FERMI
Formalization and Experimentation in the Retrieval of Multimedia Information
ESPRIT Basic Research Action n. 8134

Work Part 1

Deliverable D2:
A Logic for Information Retrieval

Consiglio Nazionale delle Ricerche
Istituto di Elaborazione della Informazione

Universitè de Grenoble Joseph Fourier
Laboratoire de Genie Informatique

Contents

List of Figures

Chapter 1

Foreward

This document presents the theoretical results of Work Part 1 of FERMI and consists of two main parts.

In the first one, Part I, the FERMI logic for Information Retrieval is presented. The logic, named MIRLOG, is the product of over 18 months of study conducted by the FERMI Team at Consiglio Nazionale delle Ricerche. Its basic features come from the partial results obtained during the first year of the Project and presented in the Deliverable D1.

Part II reports a study on user and system relevance in information retrieval, carried out by Universitè Joseph Fourier. This study places itself at a higher level of abstraction than the retrieval model given in Part I, and addresses an aspect only marginally treated in Part I.

Overall, the two Parts of the present document draw an ideal line of development: from the MIRLOG-based model, providing a very rigorous framework for dealing with the representational and inferential aspects of information retrieval, towards a more complete model, still logic-based, but also endowed with the flexibility needed to cope with the complex dynamics of relevance.

I would like to thank the researchers that have contributed to the work reported in this document: Jean-Pierre Chevallet and Marie France Bruandet from Universitè Joseph Fourier and Umberto Straccia and Fabrizio Sebastiani from Consiglio Nazionale delle Ricerche.

Carlo Meghini
Leader of the Work Part

Part I

The FERMI logic

Chapter 2

Introduction

The FERMI Project is centered around the Terminological Information Retrieval Model (Terminological Model, for short), a logic-based model that hinges on a Terminological Logic for representing documents, information needs and the relationship between them capturing a notion of relevance. Deliverable D1 reported a thorough investigation of a particular instance of the Terminological Model, namely the one based on the logic MIRTL. This investigation has highlighted negative properties of MIRTL, among which the undecidability of the main decision problems is the most relevant to the FERMI Project. At the same time, two basic *desiderata* were identified: a closed-world assumption semantics allowing to express the completeness of knowledge on certain individuals, and an implication relation which be closer to the notion of relevance than the one formalized by classical logical implication.

These three aspects provide the rationale for the main choices that have led to the definition of MIRLOG, and are illustrated in the rest of this Chapter.

2.1 From MIRTL to $\mathcal{ALC}$

No efficient system can be founded on a representation scheme suffering from undecidability. In the theory of computer science, undecidable formalisms are at the heart of fundamental notions, such as that of algorithm and that of programming language. In this respect, undecidability is a mandatory property of foundational systems, as it captures an essential aspect of computational agents. However, when it comes to the foundation of interactive systems, which is precisely the goal of FERMI, undecidability becomes an embarrassing presence. In fact, the inevitable non-termination of (sound and complete) retrieval algorithms is not only unacceptable from the user point of view, it also prevents the designers of such algorithms to perform a precise analysis on the behaviour of proposed solutions.

As a reaction to its undecidability, the concept constructors of MIRTL have been replaced by those of the logic $\mathcal{ALC}$. As a result, MIRLOGlooses number restrictions, inverse roles and singletons, while gaining full negation (hence also disjunction).

The rationale behind the choice of $\mathcal{ALC}$ is that $\mathcal{ALC}$ is a terminological logic of the same family as MIRTL, the $\mathcal{AL}$ family, that provides the typical first-order operators and can therefore be considered as a general representative of its family. More importantly, the basic decision problems for $\mathcal{ALC}$ are all decidable; in particular, testing the satisfiability of an $\mathcal{ALC}$-knowledge base is known to be PSPACE-complete.

The loss in expressivity caused by the adoption of $\mathcal{ALC}$ concept constructors for MIRLOG, is mitigated by the additional features of the latter. In particular, closure assertions offer the possibility of representing meta-information on individuals and can simulate, in most interesting cases, the information on the cardinality of role fillers expressed through number restrictions. Furthermore, assertional formulae allow to combine simple assertions in a Boolean way, so regaining part of the expressive power of the singleton operator.

2.2 Closing individuals

The classical logical implication relation, adopted by terminological logics, reflects a way of reasoning that is based on the open world assumption. Essentially, this means that a knowledge base (KB) is certain only on the status of the knowledge that it possesses, either explicitly or implicitly. What escapes its state of knowledge, is simply unknown to the KB. For instance, from the only fact that Carlo is the only known author of the document D, it does not follow that Francesco is not an author of D. In other words, under the open world assumption a KB is interpreted as a partial description that may lack information about some aspects of the modelled state of affairs.

Under certain circumstances, however, a different understanding of the contents of a document base would be in order. Typically, when indexers build document representations, they limit themselves to the specification of the positive facts, describing what the document *is,* because the specification of what the document *is not* usually amounts to an overwhelming number of negative assertions. In these cases, the lack of negative information should be understood as a tacit assertion of that information, in conformance with the principles of the closed world assumption. To stay with the previous example, in order to obtain the desired behaviour within the framework of classical implication, an indexer should specify that X is not an author of the document D for all persons X known to the document base. The material impossibility of doing that produces, as a result, an undesired behaviour of the system.

In order to solve this problem, the document representation language of the MIRLOG model has been extended with *closure assertions,* that is, assertions about the completeness of the knowledge on specified individuals. These assertions are interpreted on the basis of a non-classical semantics for $\mathcal{ALC}$, devised to this end. The extended language and its

semantics are given in Chapter 3.

2.3 Modelling the relevance of retrieval

Information Retrieval (IR) is often described in terms of relevance: its task is to find all the documents that are relevant to a given query. In logic, relevance has been the subject of numerous and fruitful investigations, which led, among other things, to the definition of several inference relations capturing different aspects of this notion. One inference relation in particular, that of tautological entailment, has been widely recognized as modelling salient features of relevance, and thereby adopted by many representation schemes, to the end of embodying a tight connection in meaning between the premises and the conclusion of any argument licensed by the underlying logic.

The semantical universe of tautological entailment is based on 4 truth values, one for each possible combination of the classical two values, hence the names of "4-valued logics" assigned to the calculi informed by this ontological view. Recently, 4-valued TLs have been studied in connection with the reasoning requirements of knowledge-based systems.

In the context of IR, we have proposed the study of a 4-valued semantics for the terminological logic supporting a model of information retrieval, given the strong relationships between relevance and tautological entailment on one hand, and relevance and information retrieval on the other. Preliminary result of these studies have been reported in D1, where a 4-valued version of the logic $\mathcal{ALC}$ has been presented, based on a semantics inspired by relevance logic and tailored to the reasoning tasks of IR. In chapter 4, this semantics is embedded in the semantics of MIRLOG, which thereby acquires the status of a 4-valued, relevance terminological logic.

2.4 The final logic

MIRLOG is described in Chapter 5. As already mentioned, its concept constructors are those of the logic $\mathcal{ALC}$. In addition, MIRLOG provides closure assertions and has a four-valued semantics. Furthermore, assertional formulae are introduced in MIRLOG to permit a more articulated expression of document descriptions.

The Gentzen-style calculus presented in Chapter 4 is extended to handle the retrieval problem on MIRLOG-knowledge bases, and its soundness and completeness are proved. Preliminary considerations on the complexity of MIRLOG and on the modelling of relevance feedabck are finally presented.

Chapter 3

Closing individuals in Terminological Logics

3.1 Introduction

The young Adso has been recently hired by the director of a very important library, supported by a sophisticate system for information retrieval (IR), based on logic. Having been educated to appreciate the high-principled spirit of logic, Adso hardly contains his enthusiasm for his new position. Moreover, thanks to a recommendation letter from his master Guglielmo, he has been given an important role, that of document indexer, even though at the beginning he will be working on very small and ordinary documents, such as letters. After studying in depth the documentation of the system, Adso tackles his first duty, that is to specify the representation of his recommendation letter.

The study of the system has revealed to Adso that letters are instances of the concept `Letter`, so, after selecting the name `d` for the one at hand, he enters into the system the assertion:

```
Letter(d).
```

The system promptly replies *Assertion added,* so Adso can proceed to specify the sender of the letter, which he does by inputing the assertion:

```
Sender(d,Guglielmo).
```

Also this time the system reacts in the expected way, and the specification of `d` goes on until Adso has entered all the available information by telling the system the corresponding assertions, some of which involving complex concepts.

At the end of the specification, Adso wants to check the result of his work by issuing some queries to the system. In the meantime, the vice-director of the library, a notorious

and ferocious enemy of logic-based IR, has approached Adso's desk, and is observing the behaviour of the new employee from Adso's back, with severe intensity. To begin the checking stage, Adso poses the entered assertions as queries, and the result to all of them is a reassuring *Yes.* Relieved by this success, Adso decides to poses more subtle queries, with the undeclared intention of impressing the observer with the power of logic and his ability to master it. Adso enters the query:

$$\neg \texttt{Book(d)}$$

asking whether `d` is not an instance of `Book`, another document type known to the system. The expected answer is *Yes,* but, to the surprise of the enquirer, the system answers *I don't know.* Adso mumbles something about the difficulties of implementing negation and then attempts a prompt recovery moving on to show the power of universal quantification. After obtaining a *Yes* in response to the query `Scottish(Guglielmo)`, he issues the query:

$$\forall \texttt{Sender.Scottish(d)},$$

asking the system whether all the senders of `d` are Scottish. As before, a *Yes* answer is expected, because Adso knows that `Guglielmo` is the only sender of `d` known to the system. But expectations are again disappointed by another *I don't know* answer. At this point, the amusement of the vice-director reaches its maximum. Adso is complimented and given to read, as a prize for his powerful argument against logic-based IR, Aristotle's *Organon,* an opus kept secret to the ordinary users of the library for its revolutionary contents.

Being not fluent in ancient Greek and also short of time, Adso sets *Organon* apart and resorts to a modern logic textbook. He then discovers that what was wrong with the previous session were his expectations. In fact, as Adso knows well, the system he is using is based on classical logical implication (in symbols, $\models$) and, from the given premises, it does not logically follow that `d` is not a letter and that all `d`'s senders are Scottish. Symbolically,

$$\begin{aligned} \texttt{D} &= \{\texttt{Letter(d)}, \texttt{Sender(d, Guglielmo)}, \texttt{Scottish(Guglielmo)}, \texttt{Book(c)}\} \\ \texttt{D} &\not\models \neg\texttt{Book(d)} \\ \texttt{D} &\not\models \forall\texttt{Sender.Scottish(d)}. \end{aligned}$$

Having learnt the basics of logical implication, Adso realizes that, in order to license the latter inference, the system must be told that there are no other senders of the letter `d`. This can be done in any TL including number restriction operators by means of the assertion $(\leq$ `1 Sender)(d)`, but, unfortunately the logic underlying the library's system does not include the $\leq$ operator; as a matter of principle, Adso refuses to add the query itself to the representation of the document, so he gives up universal quantification and focuses on the former inference. But even in this case, the only way to obtain the inference is to tell the system that `d` is not a book; but then, thinks Adso, the same problem would arise with the query:

$$\neg\texttt{Scottish(d)}.$$

At this point, Adso understands that he will obtain the expected behaviour of the system only after telling the whole story about `d`, that is not only what `d` is, but also what `d` is not. After a quick glance to the system catalog, Adso reckons that a complete description of `d` would amount to a few thousands concept assertions, namely one assertion of the form $\neg A$(`d`) for all primitive concepts A which `d` is not an instance of, and a few millions role assertions, namely $\neg R$(`d`,c) for all primitive roles R and individual constants c such that c is not an R-filler of `d`. A comprehensible desperation takes possess of Adso.

As designers of logic-based models of IR, we sympathize with Adso and would like to help him by devising a model that overcomes the illustrated problem. The proposed model extends the one used by Adso with assertions on individual constants, such as `a`, having the form:

$$\texttt{Cl(a)}.$$

Informally, an assertion of this kind means that the document base contains, whether explicitly or implicitly, everything that is true about `a`, and every other piece of information on `a` is to be understood as false. An assertion like the above one is called a *closure assertion,* as the reading of the information concerning `a` induced by the assertion is akin to a closed-world assumption. The individuals that are subject to closure assertions are said to be *closed.*

The desired effect of closure assertions is twofold. From one hand, they are meant to give the indexer the possibility of specifying meta-information, regarding the way in which the information on certain individuals is to be considered. From the other hand, they are meant to guide the inferential behaviour of the system on closed individuals in a way that reflects Adso's, and our own, intuition. More precisely, while the lack of information on non-closed individual is to be interpreted, in the usual way, as evidence of the incompleteness of the representation, the lack of information on closed individuals is to be interpreted as evidence to the contrary. Returning to the previous example, this means that the desired interpretation of closure assertions would grant the following inferences:

$$\begin{aligned}\texttt{D} \cup \{\texttt{Cl(d)}\} &\models_c \neg\texttt{Book(d)}\\ \texttt{D} \cup \{\texttt{Cl(d)}\} &\models_c \forall\texttt{Sender.Scottish(d)},\end{aligned}$$

where $\models_c$ is the inference relation of the new model. $\models_c$ should clearly be non-monotonic, as the addition of knowledge should block the application of closed-world reasoning. For instance, the following should hold:

$$\texttt{D} \cup \{\texttt{Cl(d)}\} \cup \{\texttt{Book(d)}\} \not\models_c \neg\texttt{Book(d)}.$$

In the rest of this chapter, after reviewing related work, we define knowledge bases including closure assertions and provide a semantics for them, based on the notion of

possible world. The concluding sections are devoted to the discussion of the behaviour of KBs with closure assertions. In particular, Section 3.4 illustrates formally the positive properties of the semantics, while Section 3.5 presents a form of inconsistency due to the interaction between normal and closure assertions.

The proofs of the propositions introduced in this Chapter are not given, as they are mostly re-phrasing of the corresponding proofs Chapter 5, given in the appendix.

3.2 Relation to Other Approaches

Since the seminal paper by Reiter [Reiter, 1987], many forms of closed-world assumption (CWA) have been investigated (see [Lukaszewicz, 1990] for a thorough review of this work). Without going into the details of the single proposals, we observe that none of them is able to apply the closure to specified individuals. Furthermore, the results obtained in these studies do not carry over Terminological Logics, as they are formulated for universal theories without equality. As it is well known, the first-order counterparts of Terminological Logics are first-order calculi with existential quantification [Patel-Schneider, 1987c] and even equality if number restrictions are allowed.

For this reason, formulations of the CWA for Terminological Logics have recently appeared which are based on the usage of an epistemic operator [Donini *et al.*, 1992a; Donini *et al.*, 1992b]. The basic idea behind these proposals is to enforce a CWA reading of the information about an individual `a` by using an epistemic operator **K** in queries regarding `a`. Applied to the previous example, this means that in order to obtain a positive answer on the non-membership of `d` to the `Book` concept, one has to pose the query:

$$\neg \mathbf{K}\texttt{Book(d)},$$

asking is whether `d` *is not known* to the knowledge base to be a book, which is indeed the case. Analogously, the answer to the query:

$$\forall \mathbf{K}\texttt{Sender.Scottish(d)}$$

is *Yes* because there is only one known sender of `d` and he happens to be Scottish.

As made clear by these examples, the usage of an epistemic operator in queries allows one to ask questions not only on how the modelled world is, but also on what the knowledge base knows about such world [Reiter, 1990a]. It is also evident that this usage permits to capture, among other things, some form of CWA. However, a clear connection between epistemic queries to terminological knowledge bases and any of the various CWA formulations has been established only in a very restricted case (see Theorem 5.1 in [Donini *et al.*, 1992a]); thus, strictly speaking, one cannot claim full control of how epistemic queries realize CWA.

Besides this formal argument, the adoption of the epistemic approach in our setting is problematical due to the fact that queries to document bases are not assertions, but concepts, returning the individuals that, in every interpretation, are instances of the query concept [Meghini *et al.*, 1993]. Now, let us consider the document base `E` and the query `C` defined as follows:

$$\begin{aligned} \texttt{E} &= \{\texttt{Letter(d)}, \texttt{Cl(d)}, \texttt{Letter(a)}\} \\ \texttt{C} &= \neg\texttt{Book}. \end{aligned}$$

According to our intended meaning of closure assertions, the answer to `C` in `E` should be the set {`d`}, as `d` is the only closed individual in `E`. In order to achieve this goal by means of epistemic queries, `C` should be broken into two assertion queries $\texttt{C}_1$ and $\texttt{C}_2$, given by:

$$\begin{aligned} \texttt{C}_1 &= \neg\texttt{Book(a)} \\ \texttt{C}_2 &= \neg\mathbf{K}\texttt{Book(d)}. \end{aligned}$$

The transformation is cumbersome and, when applied to nested concepts, is prone to generate a number of queries which is exponential in the number of individuals in the document base. In addition, it can be performed only once the closed individuals are known. But then, it is preferable to use closure assertions in a more direct and neat way, devising a semantics that reflects the intuition behind these assertions. And this is precisely our approach.

3.3 Knowledge Bases with Closures

Let $\mathcal{O}$ be an alphabet of symbols, called individuals, and denoted by a and b. The concepts (denoted by the letters C and D) of the MIRLOG language are those of the logic $\mathcal{ALC}$, built out of primitive concepts (denoted by the letter A) and primitive roles (denoted by the letter R), according to the following rule:

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

Roles in MIRLOG are always primitive. As customary, in the rest of this document we will omit parentheses around concepts, unless the need for disambiguation arises.

Let Δ be the *domain,* a countably infinite set of symbols, called *parameters* and denoted by p_1 and p_2, and γ a fixed injective function from $\mathcal{O}$ to Δ. An *interpretation* $\mathcal{I}$ is a total function mapping every concept to a subset of Δ and every role to a subset of $\Delta \times \Delta$, so that the following equations are satisfied:

$$\begin{aligned}
\top^{\mathcal{I}} &= \Delta \\
\bot^{\mathcal{I}} &= \emptyset \\
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} &= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &= \Delta - C^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}} &= \{p_1 \in \Delta \mid \text{for all } p_2,\ (p_1, p_2) \in R^{\mathcal{I}} \text{ implies } p_2 \in C^{\mathcal{I}}\} \\
(\exists R.C)^{\mathcal{I}} &= \{p_1 \in \Delta \mid \text{there exists } p_2,\ (p_1, p_2) \in R^{\mathcal{I}} \text{ and } p_2 \in C^{\mathcal{I}}\}
\end{aligned}$$

An *assertion* is an expression having one of the following forms:

- *concept* assertion: $C(a)$;
- *role* assertion: $R(a, b)$;
- *closure* assertion: $\texttt{Cl}(a)$, where $\texttt{Cl}$ is a special operator.

For brevity, concept and role assertions will be called *simple,* and closure assertions simply *closures.* A concept assertion is *ground* if the involved concept is primitive. Role assertions are always ground. An individual subject to a closure is called *closed.*

Satisfaction of simple assertions is defined in the standard way, i.e. an interpretation $\mathcal{I}$ satisfies $C(a)$ if and only if $\gamma(a) \in C^{\mathcal{I}}$, whereas $\mathcal{I}$ satisfies $R(a, b)$ if and only if $(\gamma(a), \gamma(b)) \in R^{\mathcal{I}}$. Finally, $\mathcal{I}$ satisfies, or is a model of, a set of assertions if it satisfies each assertion in the set.

Satisfaction of closures is defined on the basis of a notion of minimal knowledge, modelled by *epistemic interpretations.* An epistemic interpretation is a pair $(\mathcal{I}, \mathcal{W})$, where $\mathcal{I}$ is an interpretation and $\mathcal{W}$ is a set of interpretations. An epistemic interpretation satisfies a closure $\texttt{Cl}(a)$ if and only if the following two conditions hold:

1. for every primitive concept symbol A, $\gamma(a) \in A^{\mathcal{I}}$ if and only if $\gamma(a) \in A^{\mathcal{J}}$ for any $\mathcal{J} \in \mathcal{W}$;
2. for every primitive role symbol R and parameter $p \in \Delta$, $(\gamma(a), p) \in R^{\mathcal{I}}$ if and only if $(\gamma(a), p) \in R^{\mathcal{J}}$ for any $\mathcal{J} \in \mathcal{W}$.

An epistemic interpretation is a model of a set of closures if it satisfies each closure in the set.

As it will be clear in a moment, in an epistemic interpretation $(\mathcal{I}, \mathcal{W})$, the elements of $\mathcal{W}$ are supposed to represent all the possible worlds that the modelled agent (the document base) is aware of, thus forming the context in which closures are to be interpreted. In particular, if a is a closed individual, condition 1 above allows $\gamma(a)$ to belong only in the extensions of the primitive concepts that contain $\gamma(a)$ in all possible worlds. Condition 2 imposes an analogous constraint on extensions of primitive roles.

A knowledge base is a pair (Σ, Ω), where Σ is a finite set of simple assertions and Ω a finite set of closures. An interpretation $\mathcal{I}$ is a *model* of (Σ, Ω) if and only if $\mathcal{I}$ is a model of Σ and $(\mathcal{I}, \mathcal{M}(\Sigma))$ is a model of Ω, where $\mathcal{M}(\Sigma)$ are the models of Σ. It is easy to verify that, for any model of a knowledge base and closed individual a, $\gamma(a)$ is allowed in the extension of a primitive concept A just in case $A(a)$ is a logical consequence of the simple assertions, in symbols $\Sigma \models A(a)$. Analogously, a model $\mathcal{I}$ of (Σ, Ω) is a model of Σ in which, for every primitive role R, closed individual a and individual b, $(\gamma(a), \gamma(b)) \in R^{\mathcal{I}}$ if and only if $\Sigma \models R(a, b)$. In other words, closures force minimal knowledge on closed individuals, ruling out models in which these individuals show up in undue places.

A knowledge base (Σ, Ω) *logically c-implies* a simple assertion α, in symbols $(\Sigma, \Omega) \models_c \alpha$, if and only if every model of (Σ, Ω) satisfies α.

Next section investigates the main properties of the model introduced so far.

3.4 Properties of the Model

First, let us consider the example introduced in Section 3.1, having:

$$\begin{aligned} \Sigma &= \{\texttt{Letter(d)}, \texttt{Sender(d,Guglielmo)}, \texttt{Scottish(Guglielmo)}, \texttt{Book(c)}\} \\ \Omega &= \{\texttt{Cl(d)}\} \end{aligned}$$

Thanks to the closure of `d`, in all the models of (Σ, Ω), $\gamma(\texttt{d})$ only belongs to the extension of `Letter` and, as first member of a pair, to that of `Sender`. The other parameters of Δ, among which there are $\gamma(\texttt{c})$ and $\gamma(\texttt{Guglielmo})$, are free to occur in any of the extensions of the primitive concepts and roles, provided, of course, that all the assertions in Σ are satisfied. Among other things, this means that:

- in all the models of (Σ, Ω), $\gamma(\texttt{d})$ is not in the extension of `Book`, hence, as desired,

$$(\Sigma, \Omega) \models_c \neg\texttt{Book(d)};$$

- in all the models of (Σ, Ω), $\gamma(\texttt{Guglielmo})$ is the only parameter to be in the extension of `Sender` as a second member of a pair whose first element is $\gamma(\texttt{d})$; moreover, in all the models of (Σ, Ω), $\gamma(\texttt{Guglielmo})$ is in the extension of `Scottish`; it follows that, all the models of (Σ, Ω) satisfy $\forall\texttt{Sender.Scottish(d)}$, therefore:

$$(\Sigma, \Omega) \models_c \forall\texttt{Sender.Scottish(d)}.$$

Thus, in order to achieve his goal, Adso would have just to close `d`.

More generally, a closure completes the knowledge on the closed individual, granting inferences requiring negative information that normally would not be licensed. As a consequence, for every concept of the language, either the closed individual is an instance of that concept or it is an instance of the negation of that concept. This result, stated by the next Proposition, is a first evidence of the adequacy of our formalization to the declared intent.

Proposition 1 *Let* (Σ, Ω) *be a knowledge base,* $\texttt{Cl}(a) \in \Omega$ *and* $C(a)$ *a concept assertion. Then, either* $(\Sigma, \Omega) \models_c C(a)$ *or* $(\Sigma, \Omega) \models_c \neg C(a)$.

The converse of the last Proposition is given by the next one, stating that, under certain circumstances, the knowledge base behaves as if it contained a certain closure. This Proposition can be proved in the strongest form, i.e. by requiring implication of only ground assertions.

Proposition 2 *Let* (Σ, Ω) *be a KB. If for every primitive concept* A*, either* $\Sigma \models A(a)$ *or* $\Sigma \models \neg A(a)$*, then* $\mathcal{I}$ *is a model of* (Σ, Ω) *if and only if* $\mathcal{I}$ *is a model of* $(\Sigma, \Omega \cup \{\texttt{Cl}(a)\})$.

It is natural to ask how c-implication relates to classical logical implication, which is denoted as $\models$. The answer to this question comes in three steps. First, a knowledge base with no closures is equivalent to a set of simple assertions, in that the two have the same models.

Proposition 3 *Let* Σ *be a set of simple assertions. Then an interpretation* $\mathcal{I}$ *is a model of* $(\Sigma, \emptyset)$ *if and only if* $\mathcal{I}$ *is a model of* Σ.

Second, in presence of closures, c-implication extends classical implication, that is $\models \subset \models_c$. This relationship is derived in two steps: next Proposition asserts that $\models \subseteq \models_c$.

Proposition 4 *Let* (Σ, Ω) *be a knowledge base and* $C(a)$ *a simple assertion. Then* $\Sigma \models C(a)$ *implies* $(\Sigma, \Omega) \models_c C(a)$.

In order to show that $\models \neq \models_c$, it suffices to consider the knowledge base (Σ, Ω) defined at the beginning of this Section. As it has been shown, $\Sigma \not\models \neg\texttt{Book}(\texttt{d})$, whereas $(\Sigma, \Omega) \models_c \neg\texttt{Book}(\texttt{d})$.

Finally, the next Proposition shows exactly what is the inferential gain of c-implication over classical implication.

Proposition 5 *Let* (Σ, Ω) *be a knowledge base and* $\texttt{Cl}(a) \in \Omega$. *Then for each primitive concept* A,

1. $\Sigma \models A(a)$ *implies* $(\Sigma, \Omega) \models_c A(a)$;

2. $\Sigma \not\models A(a)$ *implies* $(\Sigma, \Omega) \models_c \neg A(a)$.

Conversely, if (Σ, Ω) *is satisfiable, then for each primitive concept* A,

3. $(\Sigma, \Omega) \models_c A(a)$ *implies* $\Sigma \models A(a)$;

4. $(\Sigma, \Omega) \models_c \neg A(a)$ *implies* $\Sigma \not\models A(a)$.

In fact, part 1 of the last Proposition is a special case of Proposition 4 and it has been stated in this form for symmetry with the rest of the Proposition. It states that c-implication preserves implication. Part 2 further confirms the adequacy of our model, by stating that the absence of positive, ground information on closed individuals is understood as evidence to the contrary. But the most important part of the Proposition is the converse, as it establishes the consequences of c-implication. If (Σ, Ω) is not satisfiable, everything is c-implied by it, so the case is not particularly interesting, in this context. On the contrary, if (Σ, Ω) is satisfiable, the positive, ground assertions c-implied by it are exactly those implied by Σ, whereas the negative ground assertions c-implied by it are exactly the negation of the positive, ground assertions not implied by Σ.

The last result gives us the possibility of comparing our model with Naive CWA, historically the first notion of CWA to be proposed. Naive CWA is defined for finites sets of first-order sentences without equality and whose prenex normal forms contain no existential quantifiers. If T is one such sets, then the naive closure of T, *NCWA(T),* is given by:

$$NCWA(T) = T \cup \{\neg A : T \nvdash A \text{ and } A \in HB(T)\},$$

where $HB(T)$ is the Herbrand Base of T. Given the completeness of first-order logic, we may replace $\vdash$ by $\models$. Furthermore, since we are considering theories with no function symbols, the terminological correspondent of $HB(T)$ is the set of positive ground assertions. It then follows that c-implication captures Naive CWA for theories which, although not as expressive as a first-order calculus, do provide the basic connectives and both quantifiers, in some form.

3.5 Consistency

Closures add a significant expressive power to the language for document representation, but, if not used properly, open the way to a new form of inconsistency. Let us consider the following knowledge base:

$$\begin{aligned} \Sigma &= \{\exists \texttt{Sender.Scottish(d)}\} \\ \Omega &= \{\texttt{Cl(d)}\}. \end{aligned}$$

Simple assertions in Σ state that `d` has a `Sender` without saying who it is. The knowledge on `d` is thus far from being complete. Nevertheless, the closure of `d` in Ω invites the knowledge base to assume that the knowledge on `d` is complete. Needless to say, this knowledge base is inherently inconsistent, and, as a matter of fact, it has no models. Technically, the generic model of (Σ, Ω), let it be $\mathcal{I}$, by definition should satisfy the following two requirements:

1. $\mathcal{I}$ should be a model of Σ, which means that it must have at least one pair of parameters $(\gamma(\mathtt{d}), p_1)$ in the extension of `Sender` such that p_1 is in the extension of `Scottish`;

2. in $\mathcal{I}$, the pair $(\gamma(\mathtt{d}), p_1)$ can be in the extension of `Sender` only if it so does in every model of Σ, which is not the case, whatever p_1 one considers.

These two requirements are clearly incompatible, hence (Σ, Ω) is unsatisfiable.

Technically speaking, this behaviour of closures is similar to that of the Naive closure of a theory. The latter,when applied to the theory:

$$\{P(a) \vee P(b)\},$$

yields the theory:

$$\{P(a) \vee P(b), \neg P(a), \neg P(b)\}$$

obviously inconsistent [Lukaszewicz, 1990]. However, there is a big methodological difference between our approach and NCWA, or, for that matter, all other approaches with the same goal: in MIRLOG, the closure of individuals is not something happening "behind the scene", but is explicitly required by the indexer, who has therefore full control of the situation.

In the previous example, it is evident that the indexer, closing `d` has misused the expressive power of the logic, breaking the consistency of the knowledge base as it could happen if misusing any other construct. As a matter of fact, none having a minimum familiarity with logical modelling can claim that Σ gives a complete specification of the positive knowledge on `d`, hence the closure of that individual is entirely undue.

Chapter 4

Relevance semantics

This Chapter presents a Relevant TL, $\mathcal{ALC}_4$, which is the core of the FERMI logic-based IR model, as it gives a desirable "relevance" flavour of relevance logics. In order to perform automated reasoning in this logic, a general and modular inference algorithm based on a Gentzen-style calculus is developed. Finally, a four-valued version of Craig's interpolation theorem is given, with the aim of showing that the defined entailment relation captures a close structural relationship between a knowledge base and a query. In the next Chapter, we will show how interpolants can be used to model relevance feedback. A preliminary version of $\mathcal{ALC}_4$ has been presented in deliverable D1. Proofs of the lemmas, theorems and propositions presented in this Chapter as well as in the next one, are given in Appendix B.

4.1 Introduction

IR can be presented as the task of retrieving the documents that are *relevant* to a given information need (query) [van Rijsbergen, 1986a; van Rijsbergen, 1986b]. In logical terms, IR amounts to the extraction, from a document base, of those documents d that, given a query q, make the formula $d \rightarrow q$ valid, where $\rightarrow$ is the logical implication relation of the adopted logic, aiming to capture a relation of relevance of the premise d to the conclusion q.

Among the many possible readings of the term "relevance", the one captured by *relevance logic* [Anderson and Belnap, 1975], and in particular by first-order tautological entailment, can be chosen as a promising source of inspiration to the end of incorporating a logic-based form of relevance in the inference mechanism of TLs. The underlying tenet of the criticism of relevance logicians to classical logic is that relevance of a premise to a conclusion is essential for asserting the implication between the premise and the conclusion. As a consequence of this criticism, the key concern underlying relevance logics is that of formalizing in a logic-based way a more suitable form of relevance than material implication. The notion of logical entailment can thus be considered arguably closer to the

notion of relevance needed in information retrieval than the one of material implication of classical logic.

In semantical terms, relevance is modelled by adopting a four-valued semantics [Belnap, 1977a; Belnap, 1977b; Levesque, 1984] for TLs, thus obtaining *Relevant Terminological Logics*. In a relevant TL, assertions can be not only *true* or *false* in an interpretation, but also neither true nor false (a state of affairs which is known as *unknown*), or both true and false (a state of affairs which is known as *contradiction*); hence, the resulting semantical domain provides the four truth values *true*, *false*, *unknown* and *contradiction*.

Logics of this kind have already been used in Knowledge Representation and Reasoning in order to avoid the so-called paradoxes of logical implication when reasoning on concepts and individuals. These logics have also been proven to have a generally better computational behaviour than their two-valued analogues [Levesque, 1984; Patel-Schneider, 1987a; Patel-Schneider, 1986; Patel-Schneider, 1987b; Patel-Schneider, 1989].

Unfortunately, we have observed that the adoption of the by now classical four-valued semantics [Patel-Schneider, 1987a; Patel-Schneider, 1986; Patel-Schneider, 1987b; Patel-Schneider, 1989] results in a too drastic loss of inferential capabilities for IR. In addition, the algorithms and proofs are rather complex and seem not to be sufficiently modular in order to be easily adapted to the $\mathcal{AL}$ family of TLs.

The aim of our work is to present a less restrictive four-valued semantics for TLs, which could be considered as a suitable core for IR purposes, while maintaining the desired "relevance" flavour of relevance logics. In order to perform automated reasoning on this logic, we will also present a new, general and modular inference algorithm based on a Gentzen-style calculus [Börger, 1988; Gallier, 1986; Thistlewaite *et al.*, 1988] for a four-valued variant of $\mathcal{ALC}$, which we will call $\mathcal{ALC}_4$. Apart from the fact that our semantics extends in a significant way the inferences usually achieved in four-valued TLs, we claim that our Gentzen's sequent calculus is as flexible as constraint propagation methods, proposed as an effective proof procedure in two-valued semantics (see, for example, [Donini *et al.*, 1991a]). Our calculus avoids the lack of flexibility of existing approaches (see, for example, [Patel-Schneider, 1989]). Since $\mathcal{ALC}_4$ can be considered as a good representative for $\mathcal{AL}$-languages[1], our opinion is that this framework could be also considered as a good basis for both research on reasoning algorithms and complexity analysis, in a four-valued semantics context of $\mathcal{AL}$-languages. Moreover, a four-valued version of Craig's interpolation theorem is presented. This theorem states that the defined entailment relation captures a close (structural) relationship between a knowledge base and a query. Therefore, the defined entailment relation could arguably be a good theoretical and practical basis for a logic-based approach to IR.

The rest of this chapter is organized as follows. In the next section we will give the syntax and semantics of $\mathcal{ALC}_4$. In Section 4.3 we will discuss our semantics and we will

[1]Note that the constraint propagation method was first applied to $\mathcal{ALC}$, too [Schmidt-Schauß and Smolka, 1991].

show, by means of examples, differences to standard two-valued semantics, differences to existing four-valued semantics and its suitability for IR. In Section 4.4 we will present a sequent calculus for $\mathcal{ALC}_4$ and in Section 4.5 we will present a four-valued variant of Craig's interpolation theorem. Section 4.6 gives preliminary remarks on the computational complexity of $\mathcal{ALC}_4$.

4.2 The four-valued concept language $\mathcal{ALC}_4$

In this section we present the syntax and semantics of $\mathcal{ALC}_4$[2]. For a more general presentation of the operators allowed in TLs, see [Donini *et al.*, 1992d; Nebel, 1990]. For a more extensive discussion on four-valued TLs, see [Patel-Schneider, 1987a; Patel-Schneider, 1986; Patel-Schneider, 1987b; Patel-Schneider, 1988; Patel-Schneider, 1989].

4.2.1 Syntax

We assume two disjoint alphabets of symbols, called *primitive concepts* and *primitive roles*. The letter A will always denote a primitive concept and the letter R will always denote a primitive role. Furthermore, we assume an alphabet of symbols called *individuals*, disjoint from the alphabets of primitive concepts and primitive roles. Individuals will be denoted below by a and b. The *concepts* (denoted below by C and D) and the *roles* (which in $\mathcal{ALC}_4$ are always primitive) of the language $\mathcal{ALC}_4$ are formed out of primitive concepts and roles according to the following syntax rule:

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

An *assertion* is an expression of type $C(a)$ (meaning that a is an instance of C), where a is an individual and C is an $\mathcal{ALC}_4$ concept, or an expression of type $R(a, b)$ (meaning that a is related to b by means of R), where a and b are individuals and R is an $\mathcal{ALC}_4$ role. An assertion made out by a primitive symbol is called *primitive assertion*. An assertion made out by a negated primitive symbol is called *negated primitive assertion* (note that roles are never negated). Finally, an $\mathcal{ALC}_4$ *knowledge base* is a finite set of assertions.

In the following, we use parentheses only when we need to disambiguate concept expressions. For example, we will write $(\forall R.C) \sqcap D$ to mean that the concept D is not in

[2]Although we restrict our attention to a four-valued variant of $\mathcal{ALC}$, our framework can be applied to other languages as well.

the scope of $\forall R$. In addition, the term "concept" is to be understood as $\mathcal{ALC}_4$-concept, unless otherwise stated.

4.2.2 Semantics

The formal semantics of the logic $\mathcal{ALC}_4$ is four-valued. The four truth values are the elements of $2^{\{t,f\}}$, the powerset of $\{t,f\}$, *i.e.* $\{t,f\}$, $\{\}$, $\{t\}$ and $\{f\}$. These values are best understood as epistemic states of a reasoning system about some proposition. Under this view, if the truth value of a proposition contains t, then the system has evidence to the effect – or beliefs – that the proposition is true. Similarly, if the truth value of a proposition contains f, then the system believes that the proposition is false. The truth value $\{\}$ corresponds to lack of knowledge, and the truth value $\{t,f\}$ corresponds to inconsistent knowledge.

In four-valued semantics it is possible to have inconsistent knowledge about some proposition without being totally inconsistent. This property, which is shared by other relevance logics, is touted as one of the advantages of relevance logics, especially when modelling states of knowledge.

Definition 1 *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* $a \neq b$[3]*.* ∎

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.

Definition 2 *Let* $\mathcal{I}$ *be an interpretation. The* positive extension *of a concept* C *in* $\mathcal{I}$*, written* $C^{\mathcal{I}}_{+}$*, is the set of domain elements that are known to belong to the concept, and is defined as* $\{d \in \Delta^{\mathcal{I}} : t \in C^{\mathcal{I}}(d)\}$*. The* negative extension *of a concept* C *in* $\mathcal{I}$*, written* $C^{\mathcal{I}}_{-}$*, is the set of domain elements that are known not to belong to the concept, and is defined as* $\{d \in \Delta^{\mathcal{I}} : f \in C^{\mathcal{I}}(d)\}$*. The positive and negative extension of roles are defined similarly.* ∎

[3]This restriction on individuals, called *unique name assumption*, ensures that different individuals denote different elements of the domain.

Note that, unlike standard semantics, positive and negative extensions need not to be complement of each other[4]. Domain elements that are members of neither set are not known 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 and roles 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 $\mathtt{A}\sqcap\mathtt{B}$ must be the intersection of the positive extension of $\mathtt{A}$ and $\mathtt{B}$ 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.

Definition 3 *Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ be an interpretation. The interpretation function $\cdot^{\mathcal{I}}$ has to meet the following equations for concepts: for each $d \in \Delta^{\mathcal{I}}$*

$$
\begin{array}{lll}
t \in (C \sqcap D)^{\mathcal{I}}(d) & \textit{iff} & t \in C^{\mathcal{I}}(d) \textit{ and } t \in D^{\mathcal{I}}(d) \\
f \in (C \sqcap D)^{\mathcal{I}}(d) & \textit{iff} & f \in C^{\mathcal{I}}(d) \textit{ or } f \in D^{\mathcal{I}}(d) \\
t \in (C \sqcup D)^{\mathcal{I}}(d) & \textit{iff} & t \in C^{\mathcal{I}}(d) \textit{ or } t \in D^{\mathcal{I}}(d) \\
f \in (C \sqcup D)^{\mathcal{I}}(d) & \textit{iff} & f \in C^{\mathcal{I}}(d) \textit{ and } f \in D^{\mathcal{I}}(d) \\
t \in (\neg C)^{\mathcal{I}}(d) & \textit{iff} & f \in C^{\mathcal{I}}(d) \\
f \in (\neg C)^{\mathcal{I}}(d) & \textit{iff} & t \in C^{\mathcal{I}}(d) \\
t \in (\forall R.C)^{\mathcal{I}}(d) & \textit{iff} & \forall\, e \in \Delta^{\mathcal{I}}, t \in R^{\mathcal{I}}(d,e) \textit{ implies } t \in C^{\mathcal{I}}(e) \\
f \in (\forall R.C)^{\mathcal{I}}(d) & \textit{iff} & \exists\, e \in \Delta^{\mathcal{I}}, t \in R^{\mathcal{I}}(d,e) \textit{ and } f \in C^{\mathcal{I}}(e) \\
t \in (\exists R.C)^{\mathcal{I}}(d) & \textit{iff} & \exists\, e \in \Delta^{\mathcal{I}}, t \in R^{\mathcal{I}}(d,e) \textit{ and } t \in C^{\mathcal{I}}(e) \\
f \in (\exists R.C)^{\mathcal{I}}(d) & \textit{iff} & \forall\, e \in \Delta^{\mathcal{I}}, t \in R^{\mathcal{I}}(d,e) \textit{ implies } f \in C^{\mathcal{I}}(e)
\end{array}
$$

∎

Observe that, in accordance with the intuitive meaning of the qualified existential operator $\exists$, $(\exists R.C)^{\mathcal{I}}_{+} = (\neg\forall R.\neg C)^{\mathcal{I}}_{+}$ and $(\exists R.C)^{\mathcal{I}}_{-} = (\neg\forall R.\neg C)^{\mathcal{I}}_{-}$.

The notion of subsumption between two concepts is defined in terms of their positive extensions as follows.

Definition 4 *Given two $\mathcal{ALC}_4$-concepts C and D:*

1. *C* subsumes *D (written $D \sqsubseteq C$) iff $D^{\mathcal{I}}_{+} \subseteq C^{\mathcal{I}}_{+}$, for every interpretation $\mathcal{I}$;*

[4]In two-valued standard semantics, we have: $C^{\mathcal{I}}_{+} \cap C^{\mathcal{I}}_{-} = \emptyset$ and $C^{\mathcal{I}}_{+} \cup C^{\mathcal{I}}_{-} = \Delta^{\mathcal{I}}$, or equivalently $C^{\mathcal{I}}_{-} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}_{+}$.

2. C is equivalent *to D, written $C \equiv D$, iff $C^{\mathcal{I}}_{+} = D^{\mathcal{I}}_{+}$, for every interpretation $\mathcal{I}$.* ∎

With respect to assertions, we have the following definitions.

Definition 5 *An interpretation $\mathcal{I}$* satisfies an assertion α *iff*

$$\begin{array}{rll} t \in C^{\mathcal{I}}(a^{\mathcal{I}}) & \textit{if} & \alpha = C(a) \\ t \in R^{\mathcal{I}}(a^{\mathcal{I}}, b^{\mathcal{I}}) & \textit{if} & \alpha = R(a,b) \end{array}$$

∎

Definition 6 *An interpretation $\mathcal{I}$* satisfies a knowledge base Σ *iff $\mathcal{I}$ satisfies all assertions in Σ. A knowledge base* Σ entails an assertion α, *written $\Sigma \mid\!\approx \alpha$, iff for all interpretations $\mathcal{I}$, if $\mathcal{I}$ satisfies Σ then $\mathcal{I}$ satisfies α.* ∎

The inclusion of terminological axioms in the model presented thus far can be done in a similar way as for standard two-valued TLs (see, for example, [Hollunder *et al.*, 1990; Nebel, 1990]).

4.3 Discussion of the semantics

In order to compare our four-valued semantics with the two-valued one, we will use the notation "$\models_2$" for the classical two-valued logical implication relation, and "$\sqsubseteq_2$" for the two-valued subsumption relation.

4.3.1 Soundness of four-valued semantics

In this section we will show that reasoning wrt four-valued semantics is sound wrt classical semantics. To this purpose, we will show that the set of two-valued interpretations is a (proper) subset of the set of four-valued interpretations.

Consider a four-valued interpretation $\mathcal{I}$ such that for every primitive concept A and primitive role P, the positive and negative extensions are both disjoint and exhaustive, *i.e.* $A^{\mathcal{I}}_{-} = \Delta^{\mathcal{I}} \setminus A^{\mathcal{I}}_{+}$ and $P^{\mathcal{I}}_{-} = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \setminus P^{\mathcal{I}}_{+}$. By case analysis on the language operators, it can be shown that such an interpretation is a two-valued interpretation for TLs. In fact, note that for two-valued interpretations, given a concept C and a role R, the following hold:

$$t \in C^{\mathcal{I}}(d) \text{ iff } f \notin C^{\mathcal{I}}(d)$$

$$t \in R^{\mathcal{I}}(d,e) \text{ iff } f \notin R^{\mathcal{I}}(d,e)$$

Therefore, the set of two-valued interpretations is a (proper) subset of the set of four-valued interpretations. It then follows easily the soundness of reasoning in this logic with respect to standard two-valued semantics.

Lemma 1 *Let Σ be a knowledge base, α an assertion and C, D two concepts. Then*

1. *if $C \sqsubseteq D$ then $C \sqsubseteq_2 D$;*
2. *if $\Sigma \mid\!\approx \alpha$ then $\Sigma \models_2 \alpha$.* ∎

Soundness of the subsumption relation and the entailment relation is an important requirement if the semantics is to capture some of the intuitive ideas underlying TLs.

4.3.2 Some subsumption relations

The subsumption and entailment relations supported by $\mathcal{ALC}_4$ are interesting subsets of the two-valued relations and are suitable to IR, as we will see below.

In $\mathcal{ALC}_4$, the concept

$$\texttt{Document} \sqcap \exists \texttt{dealswith.Relevance}$$

subsumes, as desired, the concept :

$$\begin{array}{lll} \texttt{Document} & \sqcap & (\exists \texttt{dealswith.(InformationRetrieval} \sqcap \texttt{Relevance})) \\ & \sqcap & (\forall \texttt{author.Italian}) \end{array}$$

i.e., a document dealing with IR and relevance and whose authors are all Italian is also a document dealing with relevance. A more complicated subsumption relationship is the one between concept C_1 given by:

$$\begin{array}{lll} \texttt{Document} & \sqcap & (\exists \texttt{dealswith.InformationRetrieval}) \\ & \sqcap & (\exists \texttt{dealswith.TerminologicalLogics}) \\ & \sqcap & (\exists \texttt{dealswith.Relevance}) \\ & \sqcap & (\forall \texttt{dealswith.LogicBased}) \\ & \sqcap & (\exists \texttt{type.Article}) \\ & \sqcap & (\forall \texttt{author.Italian}) \\ & \sqcap & (\exists \texttt{author.Researcher}) \\ & \sqcap & (\exists \texttt{typesetwith.LaTeX}) \end{array} \tag{4.1}$$

and concept C_2 given by:

$$\begin{array}{rcl} \texttt{Document} & \sqcap & (\exists \texttt{dealswith}.(\texttt{Relevance} \sqcap \texttt{LogicBased})) \\ & \sqcap & (\exists \texttt{type}.(\texttt{Article} \sqcup \texttt{Book} \sqcup \texttt{TechnicalReport})). \end{array} \tag{4.2}$$

In words, a document (a) of type article, (b) typeseted in LaTeX, (c) whose authors are Italian and at least one of them is a researcher, and (d) dealing with IR, TLs and relevance, all of these are logic-based, *is also* a document dealing with a logic-based approach to relevance, whose type is either article, or book or technical report. Therefore, if `doc324` is described by means of concept C_1, *i.e.* C_1(`doc324`) is in Σ, then `doc324` would be retrieved by query C_2. In fact, $\Sigma \not\approx C_2$(`doc324`), which is a perfectly reasonable and desirable inference for IR purposes.

4.3.3 Some entailment relations

In the following we will restrict our attention only to the entailment relation, as subsumption can be defined in terms of entailment (see next section).

Consider the assertion

$$\begin{array}{rcl} (\texttt{MultimediaDocument} & & \\ & \sqcap & (\forall \texttt{component}.\exists \texttt{doctitle}.\{\texttt{Fermi}\}) \\ & \sqcap & (\exists \texttt{component}.((\exists \texttt{doctype.Movie}) \sqcap \{\texttt{vt2}\})) \\ & \sqcap & (\exists \texttt{component}.((\exists \texttt{doctype.Text}) \sqcap \{\texttt{t2}\})) \\ & \sqcap & (\exists \texttt{component}.\{\texttt{ut2}\}) \\ & \sqcap & (\exists \texttt{component}.\{\texttt{ut3}\}))(\texttt{doc2}) \end{array} \tag{4.3}$$

saying that `doc2` is a multimedia document with at least four components: a text `t2` a movie `vt2`, and `ut2` and `ut3` of undefined type. Let us now consider the knowledge base Σ_1 defined as follows:

$$
\begin{aligned}
\Sigma_1 \;=\; \{\;\; & (\texttt{Document} \\
& \qquad \sqcap\ (\exists \texttt{dealswith.InformationRetrieval}) \\
& \qquad \sqcap\ (\exists \texttt{dealswith.TerminologicalLogics}) \\
& \qquad \sqcap\ (\exists \texttt{dealswith.Relevance}) \\
& \qquad \sqcap\ (\forall \texttt{dealswith.LogicBased}) \\
& \qquad \sqcap\ (\exists \texttt{type.Article}) \\
& \qquad \sqcap\ (\forall \texttt{author.Italian}) \\
& \qquad \sqcap\ (\exists \texttt{author.Researcher}) \\
& \qquad \sqcap\ (\exists \texttt{typesetwith.LaTeX}))(\texttt{doc324}), \\
& (\texttt{Document} \\
& \qquad \sqcap\ (\exists \texttt{dealswith.InformationRetrieval}) \\
& \qquad \sqcap\ (\exists \texttt{dealswith.TerminologicalLogics}) \\
& \qquad \sqcap\ (\forall \texttt{author.Italian}))(\texttt{doc211}), \\
& (\texttt{MultimediaDoc} \\
& \qquad\qquad \sqcap\ (\forall \texttt{component.}\exists \texttt{doctitle.}\{\texttt{Fermi}\}) \\
& \qquad\qquad \sqcap\ (\exists \texttt{component.}((\exists \texttt{doctype.Movie}) \sqcap \{\texttt{vt2}\})) \\
& \qquad\qquad \sqcap\ (\exists \texttt{component.}((\exists \texttt{doctype.Text}) \sqcap \{\texttt{t2}\})) \\
& \qquad\qquad \sqcap\ (\exists \texttt{component.}\{\texttt{ut2}\}) \\
& \qquad\qquad \sqcap\ (\exists \texttt{component.}\{\texttt{ut3}\}))(\texttt{doc2}), \\
& \texttt{author}(\texttt{doc211}, \texttt{Umberto}), \texttt{author}(\texttt{doc2}, \texttt{Umberto}), \\
& \texttt{French}(\texttt{Umberto})\ \} \cup\ \Sigma_2 \cup\ \Sigma_3
\end{aligned}
\tag{4.4}
$$

where

1. Σ_2 states that `French` and `Italian` are "disjoint" concepts[5], *i.e.* for our purpose consider only
$$\Sigma_2 = \{\neg \texttt{Italian}(\texttt{Umberto})\} \tag{4.5}$$

2. Σ_3 states that multimedia document components are all of type movie or of type text, *i.e.* for our purpose consider only
$$\Sigma_3 = \{(\exists \texttt{doctype.}(\texttt{Movie} \sqcup \texttt{Text}))(\texttt{ut2})\} \tag{4.6}$$

About modus ponens on roles

The following entailment relationships are readily verified:

$$
\begin{aligned}
\Sigma_1 \not\approx\ & (\texttt{Document} \\
& \qquad \sqcap\ (\exists \texttt{dealswith.}(\texttt{Relevance} \sqcap \texttt{LogicBased})) \\
& \qquad \sqcap\ (\exists \texttt{type.}(\texttt{Article} \sqcup \texttt{Book} \sqcup \texttt{TechnicalReport})))(\texttt{doc324})
\end{aligned}
\tag{4.7}
$$

[5] If someone is known to be French, then it is known not to be Italian.

as seen above, doc324 is a document dealing with a logic-based approach to relevance, whose type is article, book or technical report;

$$\Sigma_1 \not\approx (\texttt{MultimediaDoc} \sqcap \exists \texttt{author.Italian})(\texttt{doc2}) \tag{4.8}$$

i.e. from the fact that Umberto is the author of doc211 and that all authors of doc211 are Italian, it can be inferred that Umberto is Italian, hence that doc2 is a multimedia document with an Italian author;

$$\Sigma_1 \not\approx ((\exists \texttt{doctitle.}\{\texttt{Fermi}\}) \sqcap (\exists \texttt{doctype.Movie}))(\texttt{vt2}) \tag{4.9}$$

i.e. vt2 is retrieved as a response to the query "retrieve all movies which are titled Fermi". In fact, vt2 is a component of doc2 of type movie. All components of doc2 are titled Fermi. Therefore, vt2 is titled Fermi.

We claim that all three inferences are reasonable for IR purposes, in particular note that (4.8) and (4.9) are not trivial inferences. All three inferences are mainly based on the following key observation on our semantics, which in this case differs significantly from all other approaches. In fact, we allow *modus ponens on roles* (MPR, for short): *i.e.* for all concepts C and D, for any role R, and for all individuals a, b

$$\{(\forall R.C)(a), R(a,b)\} \not\approx C(b) \text{ and } \{(\forall R.C)(a), (\exists R.D)(a)\} \not\approx (\exists R.C \sqcap D)(a) \tag{4.10}$$

This kind of inference is not allowed by other four-valued TLs, as, for example, in [Patel-Schneider, 1987a; Patel-Schneider, 1986; Patel-Schneider, 1987b]). The key difference lies into the semantics of the $\forall$ operator. Patel-Schneider's condition in this case looks like

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

It is easy to see that, according to this semantics, there exists an interpretation $\mathcal{I}$ which satisfies Σ_1 and such that both t and f are in $\texttt{author}^{\mathcal{I}}(\texttt{doc211}^{\mathcal{I}}, \texttt{Umberto}^{\mathcal{I}})$ without being $t \in \texttt{Italian}^{\mathcal{I}}(\texttt{Umberto}^{\mathcal{I}})$, and thus, Σ_1 does not entail Italian(Umberto) hence,
$\Sigma_1 \not\not\approx (\texttt{MultimediaDoc} \sqcap \exists \texttt{author.Italian})(\texttt{doc2})$.

We claim that MPR is very useful for IR and have therefore included it in our logic.

About paradoxes of logical implication

In the following we will show what kind of inferences are not captured in our framework. The first two examples are about the so-called "paradoxes of logical implication" when reasoning on concepts and individuals.

First, note that the knowledge base Σ_1 has a "local inconsistency" in classical terms about Umberto's nationality, without being totally inconsistent. In fact, we have

$$\Sigma_1 \mid\!\approx (\texttt{Italian} \sqcap \neg\texttt{Italian})(\texttt{Umberto}) \tag{4.11}$$

and thus,

$$\Sigma_1 \mid\!\approx (\texttt{MultimediaDoc} \sqcap \exists\texttt{author.Italian})(\texttt{doc2}) \tag{4.12}$$

and

$$\Sigma_1 \mid\!\approx (\texttt{MultimediaDoc} \sqcap \exists\texttt{author.}\neg\texttt{Italian})(\texttt{doc2}). \tag{4.13}$$

This means that doc2 is retrieved in both cases since in Σ_1 there is *evidence* to the fact that doc2 is an instance off the queries MultimediaDoc$\sqcap\exists$author.Italian and MultimediaDoc$\sqcap$ $\exists$author.$\neg$Italian. On the other hand, in the document base Σ_1 there is nothing about vt2's authors. Therefore, as we would expect,

$$\Sigma_1 \not\mid\!\approx ((\exists\texttt{doctype.Text}) \sqcap (\forall\texttt{author.}\{\texttt{Umberto}\}))(\texttt{vt2}). \tag{4.14}$$

In two-valued semantics, since Σ_1 is inconsistent

$$\Sigma_1 \models_2 ((\exists\texttt{doctype.Text}) \sqcap (\forall\texttt{author.}\{\texttt{Umberto}\}))(\texttt{vt2}). \tag{4.15}$$

Clearly, this last kind of inference in not acceptable in IR: there is nothing in Σ_1 about vt2 which is relevant to the query $(\exists\texttt{doctype.Text}) \sqcap (\forall\texttt{author.}\{\texttt{Umberto}\})$.

This example shows, in a simple way, one of the advantages of a four-valued semantics: inconsistent knowledge bases (from a two-valued semantics point of view) do not entail everything.

Dually, concepts, whose extensions are always the entire domain of an interpretation, are not necessarily entailed by every knowledge base. In fact,

$$\Sigma_1 \mid\!\approx (\exists\texttt{.doctype.}(\texttt{Movie} \sqcup \texttt{Text}))(\texttt{ut2}) \tag{4.16}$$

and

$$\Sigma_1 \not\mid\!\approx (\forall\texttt{doctype.}(\texttt{Movie} \sqcup \neg\texttt{Movie}))(\texttt{ut3}) \tag{4.17}$$

which we feel both correct. In fact, (4.16) follows directly from Σ_1, whereas (4.17) holds since there is an interpretation $\mathcal{I}$, such that $e \in \Delta^{\mathcal{I}}$ and $t \in \texttt{doctype}^{\mathcal{I}}(\texttt{ut3}, e)$ and $\texttt{Movie}^{\mathcal{I}}(e) = \emptyset$. This is a state of affairs which models the fact that in Σ_1 there is no evidence about ut3's document type, *whatever* it could be.

Whereas, wrt two-valued semantics, we have

$$\Sigma_1 \models_2 (\forall \texttt{doctype}.(\texttt{Movie} \sqcup \neg \texttt{Movie}))(\texttt{ut3}) \tag{4.18}$$

To our opinion, missing this last kind of inference is important for IR purposes, since we want relevance of the premise to the conclusion.

About reasoning by cases

Finally, *case reasoning* does not work within our semantics. Consider the following knowledge base

$$\begin{aligned}\Sigma_4 \;=\; \{\;& \texttt{Document}(\texttt{doc74}), \texttt{component}(\texttt{doc74}, \texttt{t741}), \texttt{component}(\texttt{doc74}, \texttt{t742}),\\ & \texttt{translation}(\texttt{t741}, \texttt{t742}), \texttt{translation}(\texttt{t742}, \texttt{t743}),\\ & \texttt{ItalianText}(\texttt{t741}), \neg\texttt{ItalianText}(\texttt{t743})\\ \}\end{aligned}$$

The meaning of Σ_4 is: doc74 is a document with two components, t741, which is an Italian text, and t742. t743 is not an Italian text. Moreover, t742 is a translation of t741 and t743 is a translation of t742.

Consider now the assertion

$$\alpha = (\exists \texttt{component}.(\texttt{ItalianText} \sqcap \exists \texttt{translation}.\neg\texttt{ItalianText}))(\texttt{doc74})$$

Let $\mathcal{I}$ be an interpretation which satisfies Σ_4 such that $\texttt{ItalianText}^{\mathcal{I}}(\texttt{t742}^{\mathcal{I}}) = \emptyset$. It is easy to see that such an interpretation exists. Now, it follows that

$$t \notin (\exists \texttt{component}.(\texttt{ItalianText} \sqcap \exists \texttt{translation}.\neg\texttt{ItalianText}))^{\mathcal{I}}(\texttt{doc74}^{\mathcal{I}})$$

and, thus,

$$\Sigma_4 \not\models\!\!\approx \alpha \tag{4.19}$$

whereas, perhaps surprisingly,

$$\Sigma_4 \models_2 \alpha.$$

At first sight it would seem that $\Sigma_4 \not\models_2 \alpha$: since `t742` and `t741` are the only known components of `doc74`, and $\Sigma_4 \not\models_2$ (`ItalianText` $\sqcap \exists$`translation.`$\neg$`ItalianText`)(`t741`) and $\Sigma_4 \not\models_2$ (`ItalianText` $\sqcap \exists$`translation.`$\neg$`ItalianText`)(`t742`). However, consider any two-valued interpretation $\mathcal{I}$ which satisfies Σ_4. In this interpretation either `ItalianText(t742)` is true or `ItalianText(t742)` is false. In the former case, `doc74` has `t742` as a component which is an Italian text and whose non-Italian translation is `t743`. In the latter case, `doc74` has `t741` as a component which is an Italian text and whose non-Italian translation is `t742`. Therefore, in both cases α is true in $\mathcal{I}$ and, thus, $\Sigma_4 \models_2 \alpha$.

Note that $\Sigma_4 \not\mid\!\approx \alpha$ since it could be the case $\texttt{ItalianText}^{\mathcal{I}}(\texttt{t742}^{\mathcal{I}}) = \emptyset$. That is, we are uncertain about `t742`'s text language. Therefore, it is easy to see that if we know something about `t742`'s language then α holds: *i.e.*

$$\Sigma_4 \cup \{(\texttt{ItalianText} \sqcup \neg\texttt{ItalianText})(\texttt{t742})\} \mid\!\approx \alpha \tag{4.20}$$

We feel that (4.19) could be acceptable in the light of (4.20): *i.e.* since we have no relevant information about `t742`'s text language (in fact, we have no information at all about `t742`'s text language) case reasoning no longer holds.

To sum up, what kind of relevance relation is captured by $\mid\!\approx$? Roughly speaking, a knowledge base Σ entails everything that is *explicitly known*, *i.e.* that is in the transitive closure of Σ by means of MPR and the operators $\sqcap, \sqcup, \neg, \exists$, as (4.7), (4.8), (4.9), (4.12) and (4.13) demonstrate. All other inferences are left out, as (4.14), (4.17) and (4.19) show. More precisely, in order $\Sigma \mid\!\approx \alpha$ to hold, the structural components of α must have an analogue in Σ, modulo MPR. We will see this formally in Section 4.5.

4.4 A Gentzen style sequent calculus for $\mathcal{ALC}_4$

In this section we will present a complete Gentzen-like sequent calculus for entailment in $\mathcal{ALC}_4$. We will start out by arguing that the known methods for performing automated reasoning in TLs are not adequate for the present framework.

The most popular of these methods, constraint propagation, does not work because it is based on refutation, thet is it follows the schema $\alpha \models \beta$ if and only if $\alpha \wedge \neg\beta$ is unsatisfiable, which does not hold in a four truth value framework. In particular, it is easily to see that, in our four-valued semantics, every knowledge base is satisfiable.

The method developed by Patel-Schneider for four-valued TLs is not applicable in our setting. In fact, the procedures presented in [Patel-Schneider, 1987a; Patel-Schneider, 1986;

Patel-Schneider, 1987b; Patel-Schneider, 1989] are based on Levesque's method for four-valued propositional entailment [Levesque, 1984]: let

$$\alpha_i = \bigwedge_{k=1}^{n_i} \alpha_k^i$$

for $i = 1, 2$, be two propositional formulae in conjunctive normal form, where each α_k^i is a disjunct in clausal form. Then α_1 tautologically entails α_2 if and only if for each $1 \leq j \leq n_2$ there exists $1 \leq l \leq n_1$ such that $\alpha_l^1 \subseteq \alpha_j^2$.

Unfortunately, this method is not sound and complete with respect to the semantics of the $\forall$ operator; in addition, the algorithms and proofs are rather complex and seem not to be sufficiently modular in order to be easily adapted to the $\mathcal{AL}$ family of TLs.

The proposed Gentzen-style sequent calculus avoids the above problems, as we will see below.

4.4.1 A brief overview on Gentzen-style sequent calculus

Originally, Gentzen proposed [Gentzen, 1935] the calculus LK (Logisches Kalkül – Logical Calculus) for First Order Logic (FOL, for short). This calculus is based on FOL sequents, rules over sequents and axioms (which are themselves sequents). FOL sequents are expressions of the form:

$$A_1, \ldots, A_n \rightarrow B_1, \ldots, B_m$$

where $A_1, \ldots, A_n, B_1, \ldots, B_m$ are FOL formulae. The intuitive meaning of a FOL sequent is that a FOL interpretation $\mathcal{I}$ makes a FOL sequent $A_1, \ldots, A_n \rightarrow B_1, \ldots, B_m$ true iff $\mathcal{I}$ makes true the FOL formula

$$A_1 \wedge \ldots \wedge A_n \supset B_1 \vee \ldots \vee B_m$$

or its equivalent

$$\neg A_1 \vee \ldots \vee \neg A_n \vee B_1 \vee \ldots \vee B_m.$$

As it is easily seen, a sequent $\Gamma \rightarrow \Delta$ is logically equivalent to the conditional $\Gamma \supset \Delta$, however the usage of the $\rightarrow$ symbol is due to the fact that the implication connective may occur in any of the formulae in Γ or Δ.

There are two kinds of rules. One includes expressions of the form:

$$\frac{\Gamma' \rightarrow \Delta'}{\Gamma \rightarrow \Delta} \tag{4.21}$$

having the following meaning: in order to prove the sequent $\Gamma \rightarrow \Delta$ prove the equivalent sequent $\Gamma' \rightarrow \Delta'$, which is supposed to be easier to prove. The equivalence between the two sequents is based on the following property: $\Gamma \models \Delta$ if and only if $\Gamma' \models \Delta'$.

The second type of rules includes expressions of the form:

$$\frac{\Gamma' \rightarrow \Delta' \quad \Gamma'' \rightarrow \Delta''}{\Gamma \rightarrow \Delta} \tag{4.22}$$

which has the following meaning: in order to prove the sequent $\Gamma \rightarrow \Delta$ prove both the sequents $\Gamma' \rightarrow \Delta'$ and $\Gamma'' \rightarrow \Delta''$, which are again supposed to be easier to prove. Also this kind of rules are such that: $\Gamma \models \Delta$ if and only if both $\Gamma' \models \Delta'$ and $\Gamma'' \models \Delta''$.

The aim of this process of backward reasoning is to proceed until one reaches axioms, *i.e.* sequents of the form:

$$A, \Gamma \rightarrow A, \Delta.$$

An axiom is clearly a tautology.

An example of rule of type (4.21) is:

$$(\wedge \rightarrow) \quad \frac{A, B, \Gamma \rightarrow \Delta}{A \wedge B, \Gamma \rightarrow \Delta}$$

i.e. in order to prove $A \wedge B, \Gamma \rightarrow \Delta$ prove $A, B, \Gamma \rightarrow \Delta$.

An example of rule of type (4.22) is:

$$(\rightarrow \wedge) \quad \frac{\Gamma \rightarrow \Delta, A \quad \Gamma \rightarrow \Delta, B}{\Gamma \rightarrow \Delta, A \wedge B}$$

i.e. in order to prove $\Gamma \rightarrow \Delta, A \wedge B$ prove both $\Gamma \rightarrow \Delta, A$ and $\Gamma \rightarrow \Delta, B$.

A sequent is provable if, by applying rules, it can be transformed into a set of axioms. For example, the sequent:

$$A \wedge B \wedge C \rightarrow A \wedge B$$

is provable as the following "deduction tree" shows.

$$(\wedge \rightarrow) \quad \frac{(\wedge \rightarrow) \quad \dfrac{(\rightarrow \wedge) \quad \dfrac{A, B, C \rightarrow A \quad A, B, C \rightarrow B}{A, B, C \rightarrow A \wedge B}}{A \wedge B, C \rightarrow A \wedge B}}{A \wedge B \wedge C \rightarrow A \wedge B}$$

i.e. in order to prove $A \wedge B \wedge C \rightarrow A \wedge B$, prove $A \wedge B, C \rightarrow A \wedge B$. The latter sequent is provable iff $A, B, C \rightarrow A \wedge B$ is. $A, B, C \rightarrow A \wedge B$ is provable since both $A, B, C \rightarrow A$ and $A, B, C \rightarrow B$ are axioms. Therefore, $A \wedge B \wedge C \rightarrow A \wedge B$ is provable.

Gentzen provability is clearly equivalent to validity, given the semantics of sequents and the properties enjoyed by rules. In fact, the leaves of the generated deduction tree are axioms (tautologies) which are equivalent to the original sequent.

4.4.2 A calculus for entailment

We now present a calculus for entailment in $\mathcal{ALC}_4$. The main idea behind our approach is that in order to prove $\Sigma \mid\!\approx \alpha$, we attempt to prove the sequent $\Sigma \rightarrow \alpha$, where Σ is an $\mathcal{ALC}_4$-knowledge base and α is an $\mathcal{ALC}_4$-assertion. The modularity of this calculus is due to the fact that it is sufficient to provide rules for each operator of the language.

First, note that all problems listed below, which are usually considered interesting wrt TLs, are reducible to entailment.

Let Σ be a knowledge base.

Subsumption problem: Does a concept C subsume a concept D ?

Instance checking problem: Does Σ entail an assertion α ?

Realization problem: Let a be an individual occurring in Σ. The set of the *most specific concepts* of which a is an instance is defined as

$$MSC_\Sigma(a) = \{C \mid \Sigma \mid\!\approx C[a], \nexists D \text{ such that } \Sigma \mid\!\approx D[a], D \sqsubseteq C \text{ and } C \not\sqsubseteq D\}$$

What is the set of the most specific concepts of which a is an instance ?

Retrieval problem: If C is a concept, what are the individuals a such that $\Sigma \mid\!\approx C[a]$?

The task of IR can be described in terms of the retrieval problem: each document is represented by a unique identifier a which is an individual; a document base Σ is a set of assertions describing a set of documents; a query Q is a concept, *i.e.* a document description. The IR problem amounts to retrieve all documents a which are instances of the query concept Q in Σ.

As the following lemma shows, also subsumption can be defined in terms of entailment.

Lemma 2 *Let C, D be two concepts and a an individual. Then $C \sqsubseteq D$ if and only if $\{C(a)\} \mid\!\approx D(a)$.* ∎

Since the retrieval problem, as well as the realization problem, can be reduced to the instance checking problem, we will concentrate on the latter: *i.e.* the problem whether $\Sigma \not\approx \alpha$. Moreover, we can restrict our attention to those Σ and α made out of concepts in *Negation Normal Form* (NNF, for short), without losing generality.

Definition 7 *A concept is in* Negation Normal Form *iff if it contains a concept negation then it is the negation of a primitive concept.* ∎

It is easy to see that every concept can be transformed into an equivalent concept in NNF in polynomial time. Note that in $\mathcal{ALC}_4$ roles are already in NNF.

Lemma 3 *Each concept can be transformed into an equivalent NNF concept in polynomial time.* ∎

In the following we will tacitly consider only assertions involving NNF concepts.

Axioms and rules

Our calculus for entailment in $\mathcal{ALC}_4$ is defined as follows.

Assume an alphabet of symbols called *variables* (denoted below by x), disjoint from the alphabet of primitive concepts, primitive roles and individuals. The alphabet of *terms* is the union of the alphabets of variables and individuals (terms are denoted by t and t'). The interpretation function of an interpretation $\mathcal{I}$ is extended to variables by letting $\cdot^{\mathcal{I}}$ map every variable into $\Delta^{\mathcal{I}}$. Moreover, variables could appear in $\mathcal{ALC}_4$ assertions.

Definition 8 *A* sequent *is an expression of the form* $\Gamma \rightarrow \Delta$, *where* Γ *and* Δ *are respectively sequences* $\alpha_1, \ldots, \alpha_n$ $(n \geq 1)$ *and* $\beta_1, \ldots, \beta_m$ $(m \geq 1)$ *of* $\mathcal{ALC}_4$ *assertions.* Γ *is called the* antecedent *and* Δ *is called the* succedent. ∎

Our aim is to introduce a calculus such that $\Sigma \rightarrow \alpha$ is provable from the axioms and the rules of the calculus if and only if $\Sigma \not\approx \alpha$.

It should be noted that the semantics of sequents suggests that instead of using sequences, we could have used sets. We could indeed define sequents in terms of finite sets Γ, Δ of assertions. For simplicity we will use the same notation for both.

Definition 9 *A sequent* $\alpha_1, \ldots, \alpha_n \rightarrow \beta_1, \ldots, \beta_m$ *is* satisfiable *iff there is an interpretations* $\mathcal{I}$ *such that if* $\mathcal{I}$ *satisfies* $\{\alpha_1, \ldots, \alpha_n\}$ *then* $\mathcal{I}$ *satisfies some* β_j. *A sequent* $\Gamma \rightarrow \Delta$ *is* valid *iff all interpretations satisfy* $\Gamma \rightarrow \Delta$. *A sequent* $\Gamma \rightarrow \Delta$ *is called* falsifiable *iff it is not valid.* ∎

For example, the sequent $(C \sqcap D)(a) \rightarrow C(a)$ is valid, whereas the sequent $(C \sqcup D)(a) \rightarrow C(a)$ is not (*i.e.* it is falsifiable). From the above definition it follows that $\Sigma \not\approx \alpha$ if and only if $\Sigma \rightarrow \alpha$ is valid.

The rules operating on sequents fall naturally into two categories: those operating on assertions occurring in the antecedent, and those on assertions occurring in the succedent. The application of a rule may cause a sequent to be split into two sequents.

Definition 10 (Axioms) Axioms *are sequents of the form*

$$\alpha, \Gamma \rightarrow \alpha, \Delta$$

where α *is a primitive or negated primitive assertion*[6]. ∎

Every rule consists of one or two upper sequents called *premises* and of a lower sequent called *conclusion*. The rules of our calculus are then defined on NNF expressions as follows.

Definition 11 (Rules) *The* inference rules *of the sequent calculus for* $\mathcal{ALC}_4$ *are the following:*

$$(\sqcap \rightarrow) \quad \frac{C(t), D(t), \Gamma \rightarrow \Delta}{(C \sqcap D)(t), \Gamma \rightarrow \Delta}$$

$$(\rightarrow \sqcap) \quad \frac{\Gamma \rightarrow \Delta, C(t) \quad \Gamma \rightarrow \Delta, D(t)}{\Gamma \rightarrow \Delta, (C \sqcap D)(t)}$$

$$(\sqcup \rightarrow) \quad \frac{C(t), \Gamma \rightarrow \Delta \quad D(t), \Gamma \rightarrow \Delta}{(C \sqcup D)(t), \Gamma \rightarrow \Delta}$$

$$(\rightarrow \sqcup) \quad \frac{\Gamma \rightarrow \Delta, C(t), D(t)}{\Gamma \rightarrow \Delta, (C \sqcup D)(t)}$$

$$(\rightarrow \forall) \quad \frac{\Gamma, R(t, x) \rightarrow \Delta, C(x)}{\Gamma \rightarrow \Delta, (\forall R.C)(t)}$$

$$(\exists \rightarrow) \quad \frac{R(t, x), C(x), \Gamma \rightarrow \Delta}{(\exists R.C)(t), \Gamma \rightarrow \Delta}$$

[6] Note that α could be a generic assertion. The restriction on α is used only for developing easier proofs.

$$(\rightarrow \exists)\ \frac{\Gamma \rightarrow \Delta, (\exists R.C)(t), R(t,t') \quad \Gamma \rightarrow \Delta, (\exists R.C)(t), C(t')}{\Gamma \rightarrow \Delta, (\exists R.C)(t)}$$

$$(mpr \rightarrow)\ \frac{(\forall R.C)(t), R(t,t'), C(t'), \Gamma \rightarrow \Delta}{(\forall R.C)(t), R(t,t'), \Gamma \rightarrow \Delta}$$

where x is a new variable (called also eigenvariable*) which does not appear in the conclusion of the rules and t, t' are terms.* ∎

4.4.3 Provability, Soundness and Completeness

Every inference rule can be represented as a tree with two nodes if the rule has a single premise, or three nodes if the rule has two premises. In both cases, the root of the tree is labeled with the conclusion of the rule and the leaves (called sons) are labeled with the premises.

Definition 12 *A* deduction tree *is a tree whose nodes are labeled with a sequent, and is closed under the rules of Definition 11 in the following sense:*

1. *every node labeled with a sequent is a deduction tree;*
2. *for any deduction tree T' whose root is labeled $\Gamma' \rightarrow \Delta'$, for any instance of a rule with premise $\Gamma' \rightarrow \Delta'$ and conclusion $\Gamma \rightarrow \Delta$, the tree T whose root is labeled with $\Gamma \rightarrow \Delta$ and has as unique son the root of T', is a deduction tree;*
3. *for any two deduction trees T' and T'' whose roots are labeled $\Gamma' \rightarrow \Delta'$ and $\Gamma'' \rightarrow \Delta''$, respectively, for any instance of a rule with premises $\Gamma' \rightarrow \Delta'$ and $\Gamma'' \rightarrow \Delta''$ and conclusion $\Gamma \rightarrow \Delta$, the tree T whose root is labeled with $\Gamma \rightarrow \Delta$ and as as sons only the roots of T' and T'', is a deduction tree.*

The sequent labeling the root of a deduction tree is called conclusion *of the deduction tree. The* depth *of a deduction tree is defined as the length of the longest path from the root to the leaves.* ∎

Definition 13 *A* proof tree *is a deduction tree whose leaves are labeled with an axiom.* ∎

Definition 14 *A* counterexample tree *is a deduction tree such that some leaf is labeled with a falsifiable sequent.* ∎

Definition 15 *A sequent* $\Gamma \to \Delta$ *is* provable, *written* $\mid\!\sim \Gamma \to \Delta$, *iff there is a proof tree having* $\Gamma \to \Delta$ *as the conclusion.* ∎

The proof of a sequent $\Gamma \to \Delta$ proceeds backward, by constructing a proof tree with root $\Gamma \to \Delta$ to which the rules are applied until each branch reaches an axiom.

For example, the deduction tree below is a proof tree[7] with conclusion $(C \sqcup D)(a), E(a) \to (E \sqcap (C \sqcup D))(a)$ and thus $\mid\!\sim (C \sqcup D)(a), E(a) \to (E \sqcap (C \sqcup D))(a)$.

$$\dfrac{\dfrac{C(a), E(a) \to E(a) \qquad \dfrac{C(a), E(a) \to C(a), D(a)}{C(a), E(a) \to (C \sqcup D)(a)}}{C(a), E(a) \to (E \sqcap (C \sqcup D))(a)} \qquad \dfrac{D(a), E(a) \to E(a) \qquad \dfrac{D(a), E(a) \to C(a), D(a)}{D(a), E(a) \to (C \sqcup D)(a)}}{D(a), E(a) \to (E \sqcap (C \sqcup D))(a)}}{(C \sqcup D)(a), E(a) \to (E \sqcap (C \sqcup D))(a)}$$

The following proof tree, shows that $(\forall R.A \sqcap B)(a) \to (\forall R.A)(a)$ is provable.

$$\dfrac{\dfrac{\dfrac{(\forall R.A \sqcap B)(a), R(a,x), A(x), B(x) \to A(x)}{(\forall R.A \sqcap B)(a), R(a,x), (A \sqcap B)(x) \to A(x)}}{(\forall R.A \sqcap B)(a), R(a,x) \to A(x)}}{(\forall R.A \sqcap B)(a) \to (\forall R.A)(a)}$$

The following is a counterexample tree for the sequent $(C \sqcup D)(a) \to C(a)$ and it can be shown that $\not\mid\!\sim (C \sqcup D)(a) \to C(a)$.

$$\dfrac{C(a) \to C(a) \quad D(a) \to C(a)}{(C \sqcup D)(a) \to C(a)}$$

[7]Note that a deduction tree is represented upside down, *i.e.* the root is at the bottom.

In fact the sequent $D(a) \rightarrow C(a)$ is falsifiable by any interpretation $\mathcal{I}$ such that $t \in D^{\mathcal{I}}(a^{\mathcal{I}})$ and $t \notin C^{\mathcal{I}}(a^{\mathcal{I}})$; note that this also means that $\{(C \sqcup D)(a)\} \not\mid\!\approx C(a)$.

We are going now to prove the soundness of our calculus: *i.e.* if the sequent $\Gamma \rightarrow \Delta$ is provable then $\Gamma \rightarrow \Delta$ is valid. The following lemma states the soundness of axioms.

Lemma 4 *No axiom is falsifiable. Equivalently, every axiom is valid.* ∎

The soundness of the rules is established by the following lemma.

Lemma 5 *For each rule in Definition 11, the conclusion of the rule is falsifiable iff at least one of the premises of the rule is falsifiable. Equivalently, the conclusion of the rule is valid iff all premises of the rule are valid.* ∎

Using the two lemmas above, we can prove the soundness of our calculus.

Theorem 1 (Soundness) *If a sequent $\Gamma \rightarrow \Delta$ is provable, then it is valid.* ∎

A simple corollary gives the soundness of our calculus wrt entailment.

Corollary 1 *Let Σ be a knowledge base and α an assertion. Then $\mid\!\sim \Sigma \rightarrow \alpha$ implies $\Sigma \mid\!\approx \alpha$.* ∎

In order to prove the completeness of our calculus (if the sequent $\Gamma \rightarrow \Delta$ is valid then $\Gamma \rightarrow \Delta$ is provable), we define signed assertions. Let T and NT be two symbols not appearing in the considered alphabets.

Definition 16 Signed assertions of type a, b, c *and* d, *and their* components *are defined as follows (C, D are concepts, R is a role, and t, t' are terms):*

- *type-a signed assertions:*

Assertion	*Components*	
α	α_1	α_2
$T((C \sqcap D)(t))$	$T(C(t))$	$T(D(t))$
$NT((C \sqcup D)(t))$	$NT(C(t))$	$NT(D(t))$

- *type-b signed assertions:*

Assertion	*Components*	
β	β_1	β_2
$T((C \sqcup D)(t))$	$T(C(t))$	$T(D(t))$
$NT((C \sqcap D)(t))$	$NT(C(t))$	$NT(D(t))$

- *type-c signed assertions:*

Assertion	*Components*	
γ	γ_1	γ_2
$T((\forall R.C)(t))$	$T(R(t,t'))$	$T(C(t'))$
$NT((\exists R.C)(t))$	$T(R(t,t'))$	$NT(C(t'))$

- *type-d signed assertions:*

Assertion	*Components*	
δ	δ_1	δ_2
$T((\exists R.C)(t))$	$T(R(t,t'))$	$T(C(t'))$
$NT((\forall R.C)(t))$	$T(R(t,t'))$	$NT(C(t'))$.

∎

Definition 17 *Let α be an assertion. Then $T(\alpha)$ and $NT(\alpha)$ are called* conjugated assertions**.** ∎

We define then satisfaction of signed assertions as follows**.**

Definition 18 *Let $\mathcal{I}$ be an interpretation and α an assertion. Then*

$$\begin{array}{lll} \mathcal{I} \textit{ satisfies } T(\alpha) & \textit{iff} & \mathcal{I} \textit{ satisfies } \alpha \\ \mathcal{I} \textit{ satisfies } NT(\alpha) & \textit{iff} & \mathcal{I} \textit{ does not satisfy } \alpha. \end{array}$$

∎

Definition 19 *Let $\mathcal{I}$ be an interpretation. The* extended language wrt $\mathcal{I}$ *is obtained by adding to the set of individuals a set of new constants, one constant* $\mathbf{d}$ *for each element d of $\Delta^{\mathcal{I}}$. The interpretation function $\cdot^{\mathcal{I}}$ is extended to the new constants by defining:*

$$\mathbf{d}^{\mathcal{I}} = d.$$

∎

Lemma 6 *Let $\mathcal{I}$ be an interpretation. Then,*

1. *for any signed assertion α of type a, $\mathcal{I}$ satisfies α iff $\mathcal{I}$ satisfies both α_1 and α_2;*
2. *for any signed assertion β of type b, $\mathcal{I}$ satisfies β iff $\mathcal{I}$ satisfies β_1 or β_2;*
3. *for any signed assertion γ of type c, $\mathcal{I}$ satisfies γ iff if $\mathcal{I}$ satisfies γ_1 then $\mathcal{I}$ satisfies γ_2, for every* **d** *such that $d \in \Delta^{\mathcal{I}}$;*
4. *for any signed assertion δ of type d, $\mathcal{I}$ satisfies δ iff $\mathcal{I}$ satisfies δ_1 and δ_2, for at least one* **d** *such that $d \in \Delta^{\mathcal{I}}$.* ∎

In view of Lemma 6, signed assertions of type a are also called assertions of *conjunctive type*, signed assertions of type b are called assertions of *disjunctive type*, signed assertions of type c are called assertions of *universal type* and signed assertions of type d are called assertions of *existential type*.

Definition 20 *Let H be a subset of terms. A* Hintikka set S wrt H *is a set of signed assertions such that the following conditions hold for all signed assertions α, β, γ, δ of type a,b,c,d, respectively:*

1. *H0: No conjugated primitive assertions or conjugated negated primitive assertions are in S;*
2. *H1: If a type-a assertion α is in S, then both α_1 and α_2 are in S;*
3. *H2: If a type-b assertion β is in S, then either β_1 is in S or β_2 is in S;*
4. *H3: If a type-c assertion γ is in S, then for every term $t' \in H$, if γ_1 is in S then γ_2 is in S;*
5. *H4: If a type-d assertion δ is in S, then there is at least one term $t' \in H$ such that both δ_1 and δ_2 are in S.* ∎

Lemma 7 *Every Hintikka set S wrt a set of terms H is satisfiable in an interpretation $\mathcal{I}$ with domain H.* ∎

In order to prove the completeness, our goal is, given a sequent $\Gamma \rightarrow \Delta$, to attempt to falsify it. For this, we will describe the procedure **Search** (see Figures 4.1 and 4.2) with the following properties:

1. if the original sequent is valid, the **Search** procedure stops after a finite number of steps, yielding a proof tree;
2. if the original sequent is falsifiable, the **Search** procedure constructs a possibly infinite deduction tree, and along some (possibly infinite) path in the tree it can be shown that a Hintikka set exists, which yields a counterexample for the sequent.

procedure Search($\Gamma \to \Delta$: sequent; **var** T: tree);
  begin
    Let T be the one-node tree labeled with $\Gamma \to \Delta$;
    Let $Terms_{Used} := \langle (t_1, nil), \ldots, (t_n, nil) \rangle$ and
    $Terms_{Avail} := \langle x_1, \ldots, x_m, \ldots \rangle$ with $t_1, \ldots, t_n$ as explained.
    while not all leaves of T are finished **do**
      $TERM_1 := Terms_{Used}$; $T_0 := T$; $FORM_1 := FORM_{Used}$
      for each leaf node of T_0 **do**
        if not finished(node) **then** Expand(node, T);
      $Terms_{Used} := TERM_1$;
      $FORM_{Used} := FORM_1$;
    endwhile;
    if all leaves are axioms **then write**("T is a proof tree of $\Gamma \to \Delta$")
    else write("$\Gamma \to \Delta$ is falsifiable")
  end

Figure 4.1: The **Search** procedure

Roughly, **Search** builds a deduction tree, starting from $\Gamma \to \Delta$, applying the rules of Definition 11, using some lists of terms for handling the x's and t''s in rules $(\to \forall)$, $(\exists \to)$, $(\to \exists)$ and $(mpr \to)$.

Let

$$Terms_{Used} = \langle t_1, \ldots, t_n \rangle,$$

be the list of terms occurring in $\Gamma \to \Delta$, and let

$$Terms_{Avail} = \langle x_1, \ldots, x_m, \ldots \rangle,$$

be a countable infinite list of variables not occurring in $\Gamma \to \Delta$.

The terms in $Terms_{Used}$ and the variables in $Terms_{Avail}$ are used as the t''s and x's for the application of rules $(\to \forall)$, $(\exists \to)$, $(\to \exists)$ and $(mpr \to)$.

As the search of a counterexample of $\Gamma \to \Delta$ progresses, **Search** keeps track step by step of which of $\langle t_1, \ldots, t_n, x_1, x_2, x_3, \ldots \rangle$ have been thus far activated. The list of activated terms is kept in $Terms_{Used}$ and the list of available variables in $Terms_{Avail}$.

Every time a rule $(\to \forall)$ or $(\exists \to)$ is applied, as variable x, **Search** uses the head of the list $Terms_{Avail}$, appends x to the end of $Terms_{Used}$ and deletes x from the head of $Terms_{Avail}$.

procedure Expand(node: tree-address; **var** T: tree);
begin
Let $s = \alpha_1, \ldots, \alpha_n \rightarrow \beta_1, \ldots, \beta_m$ be the label of the node;
Let S be the one-node tree labeled with $\alpha_1, \ldots, \alpha_n \rightarrow \beta_1, \ldots, \beta_m$;
for $i := 1$ to n **do**
if nonatomic(α_i) **then** GrowLeft(α_i, S, s);
for $i := 1$ to m **do**
if nonatomic(β_i) **then** GrowRight(β_i, S);
T:= dosubstitution(T,node,S)
end

procedure GrowLeft(A: assertion; **var** S: tree, s: sequent);
begin
caseA of
$(C \sqcap D)(t)$,
$(C \sqcup D)(t)$: extend every nonaxiom leaf of S using the
left rule corresponding to the main connective;
$(\forall R.C)(t)$: **for** every term $t_i \in t(Terms_{Used})$
such that $R(t, t_i) \in s$ and $A \notin FORM_{Used}(i)$ **do**
extend every nonaxiom leaf of S by applying
the $(mpr \rightarrow)$ in conjunction with $R(t, t_i)$;
$FORM_1(i) := append(FORM_1(i), A)$
endfor;
$(\exists R.C)(t)$: extend every nonaxiom leaf of S by applying the $(\exists \rightarrow)$ rule
using $x = head(TERMSAvail)$ as the new variable;
$TERM_1 := append(TERM_1, (x, nil))$;
$TERMS_{Avail} := tail(TERMS_{Avail})$
endcase
end

procedure GrowLeft(A: assertion; **var** S: tree);
begin
caseA of
$(C \sqcap D)(t)$,
$(C \sqcup D)(t)$: extend every nonaxiom leaf of S using the
right rule corresponding to the main connective;
$(\exists R.C)(t)$: **for** every term $t_i \in t(Terms_{Used})$ such that $A \notin FORM_{Used}(i)$ **do**
extend every nonaxiom leaf of S by applying the $(\rightarrow \exists)$
rule using term t_i as one of the terms of the rule;
$FORM_1(i) := append(FORM_1(i), A)$
endfor;
$(\forall R.C)(t)$: extend every nonaxiom leaf of S by applying the $(mpr \rightarrow)$
rule using $x = head(TERMSAvail)$ as the new variable;
$TERM_1 := append(TERM_1, (x, nil))$;
$TERMS_{Avail} := tail(TERMS_{Avail})$
endcase
end

Figure 4.2: The auxiliary procedures for **Search**

When a rule $(\rightarrow \exists)$ is applied, **Search** uses as terms t' *each* of the terms $t_1, \ldots, t_q$ in $Terms_{Used}$ that *have not previously served as terms* t' *for that rule with same instantiation.*

In order to handle the $(\rightarrow \exists)$ rule and the $(mpr \rightarrow)$ rule correctly, it is necessary to keep track of the assertions $(\exists R.C)(t), (\forall R.C)(t)$ and $R(t, t_i)$ for which the term t_i was used. The main reason is that if a sequent has the property that all assertions in it are either primitive or negated primitive or of the form $(\exists R.C)(t)$ or $(\forall R.C)(t)$ such that all terms in $Terms_{Used}$ have already been used as terms for the above rules and if this sequent is not an axiom, then it will never become an axiom and we can stop expanding it.

Hence, we structure $Terms_{Used}$ as a list of records where every record $(t_i, FORM_{Used}(i))$ contains two fields: t_i is a term and $FORM_{Used}(i)$ a list of assertions $(\forall R.C)(t)$ (or $(\exists R.C)(t)$) for which t_i was used as a term t' for the rule $(mpr \rightarrow)$ (or $(\rightarrow \exists)$). Initially, each list $FORM_{Used}(i)$ is the null list. The lists $FORM_{Used}(i)$ are updated each time a term t' is used in a rule $(mpr \rightarrow)$ or $(\rightarrow \exists)$. We also let $t(Terms_{Used})$ denote the set of terms $\{t_i : (t_i, FORM_{Used}(i)) \in Terms_{Used}\}$.

Finally, a leaf of the tree constructed by the **Search** procedure is called *finished* if and only if either the sequent labeling it is an axiom, or the sequent contains only primitive assertions or assertions $(\forall R.C)(t)$ (or $(\exists R.C)(t)$) belonging to all of the lists $FORM_{Used}(i)$ for all $(t_i, FORM_{Used}(i)) \in Terms_{Used}$.

In order to prove the following Lemma 9 we need the so-called König's lemma.

Lemma 8 (König's lemma) *If a tree T with infinite nodes is finite branching, then there is some infinite path in T.* ∎

Lemma 9 *The* **Search** *procedure satisfies the following conditions:*

1. *If the input sequent $\Gamma \rightarrow \Delta$ is valid, then the procedure* **Search** *halts with a finite closed tree T which is a proof tree for $\Gamma \rightarrow \Delta$.*

2. *If the input sequent $\Gamma \rightarrow \Delta$ is falsifiable, either* **Search** *halts with a finite counterexample tree T and $\Gamma \rightarrow \Delta$ is falsifiable in an interpretation with finite domain, or* **Search** *generates an infinite tree T and $\Gamma \rightarrow \Delta$ is falsifiable in an interpretation with a countably infinite domain.* ∎

As a consequence we obtain the completeness theorem.

Theorem 2 (Completeness) *If a sequent $\Gamma \rightarrow \Delta$ is valid, then it is provable.* ∎

Corollary 2 *Let Σ be a knowledge base and α an assertion. Then $\Sigma \mathrel{|\!\sim} \alpha$ if and only if $\Sigma \not\approx \alpha$.* ∎

4.5 Craig's "relevant" interpolation theorem

In this section we will present a four-valued version of Craig's interpolation theorem and elaborate on its significance to IR.

Craig's interpolation theorem for FOL [Gallier, 1986] provides an answer to the following problem: given a valid FOL formula $d \supset q$, is there a FOL formula γ (the interpolant of $d \supset q$) such that $d \supset \gamma$ and $\gamma \supset q$ are valid, and γ is "structurally similar" to both d and q ? If the answer is yes then γ can be seen as an explanation about the retrieval of the document d in response to the query q, because it provides the *relevant information* which are in d for it to imply q. For example, the interpolant of $A \wedge B \wedge C \supset (A \vee D) \wedge (C \vee D \vee E)$ is $A \wedge C$. Note that not every valid formula has an interpolant. For example, $(A \supset A) \supset (B \supset B)$ is valid, and yet, no interpolant γ can be found for it.

In order to prove a four-valued version of Craig's interpolation theorem, we need some definitions.

Definition 21 *An* assertional formula *is defined inductively as follows:*

1. *every assertion is an assertional formula (called* primitive assertional formula*);*
2. *if α and β are assertional formulae,then $\alpha \wedge \beta$ and $\alpha \vee \beta$ are assertional formulae.* ∎

Satisfiability is extended to non-primitive assertional formulae as follows.

Definition 22 *Let $\mathcal{I}$ be an interpretation.*

1. $\mathcal{I}$ satisfies *an non-primitive assertional formula $\alpha \wedge \beta$ iff $\mathcal{I}$ satisfies both α and β;*
2. $\mathcal{I}$ satisfies *an non-primitive assertional formula $\alpha \vee \beta$ iff $\mathcal{I}$ satisfies α or β.* ∎

Definition 23 *A* sequent formula *is a sequent in which the antecedent and the succedent are sequences of assertional formulae.* Satisfiability, validity *and* falsifiability *of sequent formulae are defined in the same way as for sequents.* ∎

Now we are able to define interpolant in our framework.

Definition 24 *Let $\Gamma \rightarrow \Delta$ be a valid sequent. Then an* interpolant *of $\Gamma \rightarrow \Delta$ is an assertional formula γ such that*

1. *$\Gamma \rightarrow \gamma$ is a valid sequent formula;*
2. *$\gamma \rightarrow \Delta$ is a valid sequent formula;*

3. *every primitive concept, primitive role, individual and variable occurring in* γ *also occurs both in* Γ *and* Δ. ∎

The main theorem follows.

Theorem 3 (Four-valued interpolation theorem) $\Gamma \rightarrow \Delta$ *is a valid sequent iff there exists a constructible interpolant* γ *of* $\Gamma \rightarrow \Delta$. ∎

A direct consequence of this theorem is the following corollary.

Corollary 3 *Let* Σ *be a knowledge base and* α *an assertion. If there is no common role symbol or concept symbol between* Σ *and* α*, then* $\Sigma \not\approx \alpha$. ∎

Note that the corresponding corollary wrt FOL does not hold. For example, in FOL, $A \models (B \vee \neg B)$ is valid notwithstanding there are no common predicate symbols between A and $(B \vee \neg B)$. On the other hand, the corresponding corollary is true for the four-valued logics described in [Levesque, 1984; Patel-Schneider, 1987a; Patel-Schneider, 1986; Patel-Schneider, 1987b; Patel-Schneider, 1989]

Example 1 *Consider the valid sequent* $(\forall R.A)(a) \rightarrow (\forall R.B)(a), (\exists R.A)(a)$ *and the following proof tree.*

$$\dfrac{\dfrac{\dfrac{(\forall R.A)(a), R(a,x), A(x) \rightarrow B(x), R(a,x), (\exists R.A)(a) \quad (\forall R.A)(a), R(a,x), A(x) \rightarrow B(x), A(x), (\exists R.A)(a)}{(\forall R.A)(a), R(a,x), A(x) \rightarrow B(x), (\exists R.A)(a)}}{(\forall R.A)(a), R(a,x) \rightarrow B(x), (\exists R.A)(a)}}{(\forall R.A)(a) \rightarrow (\forall R.B)(a), (\exists R.A)(a)}$$

The constructed interpolant is $(\forall R.A)(a)$ *which is obtained from the transformation of* $R(a,x) \wedge A(x)$ *as described in case rule* $(\rightarrow \forall)$*in the proof of Theorem 3. For the valid sequent* $((\forall R.B \sqcap D) \sqcup (\forall R.C \sqcap D))(a) \rightarrow (\forall R.(B \sqcup C) \sqcap D)(a)$*, according toTheorem 3, the following interpolant can be build:*

$$(\forall R.B \sqcap D)(a) \vee (\forall R.C \sqcap D)(a)$$

∎

This shows that our entailment relation $\not\approx$ captures a "structural" relationship between a document base Σ and a query α and thus is a good theoretical basis for a logic-based approach to IR.

4.6 Remarks on computational complexity

In this section we present some preliminary observations on the computational complexity of the instance checking problem for $\mathcal{ALC}_4$.

Since $\mathcal{ALC}_4$ contains the operator for conjunction ($\sqcap$), disjunction ($\sqcup$), negation ($\neg$) and since propositional tautological entailment, wrt standard four-valued semantics, is co-NP-Complete [Patel-Schneider, 1987a], it follows easily that:

Theorem 4 *The problem of determining the validity of a sequent* $\Gamma \rightarrow \Delta$ *is co-NP-Hard.* ∎

As a consequence,

Corollary 4 *The subsumption, instance checking, realization and retrieval problems for* $\mathcal{ALC}_4$ *are all co-NP-Hard.* ∎

In Lemma 9 we mentioned that the **Search** algorithm could generate an infinite tree. In fact, it can be shown that the application of the calculus to the sequent $B(a) \rightarrow (\exists R.\forall R.A)(a)$ generates the infinite tree:

$$\dfrac{\dfrac{\vdots}{B(a) \rightarrow \Delta, R(a,a)} \qquad \dfrac{\dfrac{\dfrac{\vdots}{B(a), R(a,x) \rightarrow \Delta, A(x), R(a,x)} \qquad \dfrac{\vdots}{B(a), R(a,x) \rightarrow \Delta, A(x), (\forall R.A)(x)}}{B(a), R(a,x) \rightarrow \Delta, A(x)}}{B(a) \rightarrow \Delta, (\forall R.A)(a)}}{B(a) \rightarrow \Delta}$$

where $\Delta = (\exists R.\forall R.A)(a)$ Each time a new variable x is introduced by the $(\rightarrow \forall)$ rule, x can be used as term t' in the application of the $(\rightarrow \exists)$ rule applied to Δ, yielding an infinite generation of variables.

Fortunately, this proliferation of variables can be avoided by adopting more sophisticated halting criteria, described in Appendix B. In fact,

Theorem 5 *Determining the validity of a sequent* $\Gamma \rightarrow \Delta$ *is decidable.* ∎

The problem whether or not the validity problem is in NP is open.

Chapter 5

The FERMI logic

This Chapter merges the results presented in the previous two Chapters into the final (single-media, non-probabilistic) FERMI logic MIRLOG, obtained by extending the TL $\mathcal{ALC}$ with individual closures, assertional formulae, and by providing a four-valued, relevance-based semantics to the so obtained language[1]. In order to carry out effective reasoning in MIRLOG, the proof system presented in the previous chapter for the logic $\mathcal{ALC}_4$ is extended to handle closure assertions.

5.1 The concept language MIRLOG

In this section we present the syntax and semantics of MIRLOG. From the syntactical point of view, MIRLOG extends $\mathcal{ALC}_4$ with role negation and assertional formulae. Semantically, closure assertions and assertional formulae are given interpretation in a 4-valued framework. For the sake of readability, the complete syntax and semantics are presented.

5.1.1 Syntax

We assume two disjoint alphabets of symbols, called *primitive concepts* and *primitive roles*. The letter A will always denote a primitive concept and the letter P will denote a primitive role. The *concepts* (denoted below by C and D) and the *roles* (denoted below by R) of the language MIRLOG are formed out of primitive concepts and roles according to the following syntax rules:

[1]Technically, the name of the logic should be something like "$\mathcal{ALC}_4^c$", so to follow the naming conventions adopted in the TL community. However, partly because of the awkwardness of the candidate official names, partly because we would like to emphasize the context in which the logic has been developed, we prefer to name the logic "MIRLOG".

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

$$
\begin{array}{rcrl}
R & \longrightarrow & P \mid & \text{(primitive role)} \\
 & & \neg P \mid & \text{(primitive role negation)}
\end{array}
$$

Furthermore, we assume an alphabet $\mathcal{O}$ of symbols called *individuals*, disjoint from the alphabets of primitive concepts and primitive roles. Individuals will be denoted by a and b.

An *assertion* is an expression of type $C(a)$ (meaning that a is an instance of C), where a is an individual and C is a MIRLOG concept, or an expression of type $R(a, b)$ (meaning that a is related to b by means of R), where a and b are individuals and R is a MIRLOG role. An assertion made out by a primitive symbol is called *primitive assertion*. An assertion made out by a negated primitive symbol is called *negated primitive assertion*.

Assertional formulae (denoted by γ and δ) of the language MIRLOG are formed out of assertions (denoted below by α) according to the following syntax rule:

$$
\begin{array}{rcrl}
\gamma, \delta & \longrightarrow & \alpha \mid & \text{(assertion)} \\
 & & \gamma \wedge \delta \mid & \text{(assertional conjunction)} \\
 & & \gamma \vee \delta \mid & \text{(assertional disjunction)} \\
 & & \sim \gamma \mid & \text{(assertional negation)}
\end{array}
$$

An *ABox* is a finite set of assertional formulae. A *closure* is an expression of type $\texttt{Cl}(a)$, in which a is said to be *closed*. A *CBox* is a finite set of closures. An MIRLOG *knowledge base* is pair (Σ, Ω), where Σ is an ABox and Ω is a CBox. Note that Σ will be interpreted as conjunction of assertional formulae.

Finally, let ν be a new symbol, called *query parameter*, not appearing in any of the alphabets introduced so far. A *parametric assertion*, written $[\alpha](\nu)$, is an assertion in which ν can appear as an individual. A *parametric assertional formula*, written $[\gamma](\nu)$, is an assertional formula in which ν can appear as an individual. A *query*, written $[Q](\nu)$, is a finite set of parametric assertional formulae, which will be interpreted as the elements of a disjunction.

With $[\alpha](a)$, $[\gamma](a)$ and $[Q](a)$ we intend the assertion, the assertional formula and the set of assertional formulae, respectively, in which the query parameter ν is replaced by the individual a. Since the query parameter not necessarily appears in a parametric assertion, it could be the case that $[\alpha](\nu) = [\alpha](a)$, for each individual a: *e.g.* let α be $A(b)$.

5.1.2 Semantics

The formal semantics of the logic MIRLOG is based on a four-valued semantics. The four truth values are the elements of $2^{\{t,f\}}$, the powerset of $\{t,f\}$, *i.e.* $\{t,f\}$, $\{\}$, $\{t\}$ and $\{f\}$. These values are known as *contradiction*, *unknown*, *true* and *false*.

Let Δ be the *domain,* a countably infinite set of symbols, called *parameters* and denoted by p and p', and γ a fixed injective function from $\mathcal{O} \cup \{\nu\}$ to Δ.

Definition 25 *An* interpretation $\mathcal{I}$ *is a total function such that*

1. $\mathcal{I}$ *maps every concept into a function from* Δ *to* $2^{\{t,f\}}$*;*
2. $\mathcal{I}$ *maps every role into a function from* $\Delta \times \Delta$ *to* $2^{\{t,f\}}$*.* ∎

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.

Definition 26 *Let* $\mathcal{I}$ *be an interpretation. The* positive extension *of a concept* C *in* $\mathcal{I}$*, written* $C^{\mathcal{I}}_{+}$*, is the set of domain elements that are known to belong to the concept, and is defined as* $\{p \in \Delta : t \in C^{\mathcal{I}}(p)\}$*. The* negative extension *of a concept* C *in* $\mathcal{I}$*, written* $C^{\mathcal{I}}_{-}$*, is the set of domain elements that are known not to belong to the concept, and is defined as* $\{p \in \Delta : f \in C^{\mathcal{I}}(p)\}$*. The positive and negative extension of roles are defined similarly.* ∎

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

Definition 27 *Let* $\mathcal{I}$ *be an interpretation. The interpretation* $\mathcal{I}$ *has to meet the following equations for concepts: for each* $p \in \Delta$

$$
\begin{array}{lll}
t \in (C \sqcap D)^{\mathcal{I}}(p) & \textit{iff} & t \in C^{\mathcal{I}}(p) \textit{ and } t \in D^{\mathcal{I}}(p) \\
f \in (C \sqcap D)^{\mathcal{I}}(p) & \textit{iff} & f \in C^{\mathcal{I}}(p) \textit{ or } f \in D^{\mathcal{I}}(p) \\
t \in (C \sqcup D)^{\mathcal{I}}(p) & \textit{iff} & t \in C^{\mathcal{I}}(p) \textit{ or } t \in D^{\mathcal{I}}(p) \\
f \in (C \sqcup D)^{\mathcal{I}}(p) & \textit{iff} & f \in C^{\mathcal{I}}(p) \textit{ and } f \in D^{\mathcal{I}}(p) \\
t \in (\neg C)^{\mathcal{I}}(p) & \textit{iff} & f \in C^{\mathcal{I}}(p) \\
f \in (\neg C)^{\mathcal{I}}(p) & \textit{iff} & t \in C^{\mathcal{I}}(p) \\
t \in (\forall R.C)^{\mathcal{I}}(p) & \textit{iff} & \forall p' \in \Delta, t \in R^{\mathcal{I}}(p,p') \textit{ implies } t \in C^{\mathcal{I}}(p') \\
f \in (\forall R.C)^{\mathcal{I}}(p) & \textit{iff} & \exists p' \in \Delta, t \in R^{\mathcal{I}}(p,p') \textit{ and } f \in C^{\mathcal{I}}(p') \\
t \in (\exists R.C)^{\mathcal{I}}(p) & \textit{iff} & \exists p' \in \Delta, t \in R^{\mathcal{I}}(p,p') \textit{ and } t \in C^{\mathcal{I}}(p') \\
f \in (\exists R.C)^{\mathcal{I}}(p) & \textit{iff} & \forall p' \in \Delta, t \in R^{\mathcal{I}}(p,p') \textit{ implies } f \in C^{\mathcal{I}}(p') \\
t \in (\neg P)^{\mathcal{I}}(p,p') & \textit{iff} & f \in P^{\mathcal{I}}(p,p') \\
f \in (\neg P)^{\mathcal{I}}(p,p') & \textit{iff} & t \in P^{\mathcal{I}}(p,p')
\end{array}
$$

Moreover, $\top_+^{\mathcal{I}} = \Delta$, $\top_-^{\mathcal{I}} = \emptyset$, *and* $\bot_+^{\mathcal{I}} = \emptyset$, $\bot_-^{\mathcal{I}} = \Delta$. ∎

Observe that, in accordance with the intuitive meaning of the qualified existential operator $\exists$, $(\exists R.C)_+^{\mathcal{I}} = (\neg\forall R.\neg C)_+^{\mathcal{I}}$ and $(\exists R.C)_-^{\mathcal{I}} = (\neg\forall R.\neg C)_-^{\mathcal{I}}$.

The notion of subsumption between two concepts is defined in terms of their positive extensions as follows. Given two MIRLOGconcepts C and D: C *subsumes* D (written $D \sqsubseteq C$) iff $D_+^{\mathcal{I}} \subseteq C_+^{\mathcal{I}}$, for every interpretation $\mathcal{I}$; C is *equivalent* to D, written $C \equiv D$, iff $C_+^{\mathcal{I}} = D_+^{\mathcal{I}}$, for every interpretation $\mathcal{I}$. $\sqsubseteq, \equiv$ are extended to roles in a straightforward way.

With respect to assertions, we have the following definitions. An interpretation $\mathcal{I}$ *satisfies an assertion* α iff $t \in C^{\mathcal{I}}(\gamma(a))$ in case $\alpha = C(a)$, whereas $t \in R^{\mathcal{I}}(\gamma(a), \gamma(b))$ in case $\alpha = R(a, b)$. Moreover, $\mathcal{I}$ *f-satisfies an assertion* α iff $f \in C^{\mathcal{I}}(\gamma(a))$ in case $\alpha = C(a)$, whereas $f \in R^{\mathcal{I}}(\gamma(a), \gamma(b))$ in case $\alpha = R(a, b)$.

Satisfiability is now easily extended to assertional formulae as follows.

Definition 28 *Let* $\mathcal{I}$ *be an interpretation.*

1. $\mathcal{I}$ satisfies *an assertional formula* $\gamma \wedge \delta$ *iff* $\mathcal{I}$ *satisfies both* γ *and* δ;
2. $\mathcal{I}$ f-satisfies *an assertional formula* $\gamma \wedge \delta$ *iff* $\mathcal{I}$ *f-satisfies* γ *or* $\mathcal{I}$ *f-satisfies* δ;
3. $\mathcal{I}$ satisfies *an assertional formula* $\gamma \vee \delta$ *iff* $\mathcal{I}$ *satisfies* γ *or* $\mathcal{I}$ *satisfies* δ;
4. $\mathcal{I}$ f-satisfies *an assertional formula* $\gamma \vee \delta$ *iff* $\mathcal{I}$ *f-satisfies both* γ *and* δ;
5. $\mathcal{I}$ satisfies *an assertional formula* $\sim\gamma$ *iff* $\mathcal{I}$ *f-satisfies* γ;
6. $\mathcal{I}$ f-satisfies *an assertional formula* $\sim\gamma$ *iff* $\mathcal{I}$ *satisfies* γ. ∎

Satisfiability is trivially extended to parametric assertions and parametric assertional formulae[2].

Given two MIRLOG assertional formulae γ and δ, γ is *equivalent* to δ, written $\gamma \equiv \delta$, iff for every interpretation $\mathcal{I}$, $\mathcal{I}$ satisfies γ iff $\mathcal{I}$ satisfies δ.

Finally, $\mathcal{I}$ *satisfies* (is a *model* of) an ABox Σ iff $\mathcal{I}$ satisfies all assertional formulae in Σ, whereas $\mathcal{I}$ *satisfies* (is a *model* of) a query $[Q](\nu)$ iff $\mathcal{I}$ satisfies some parametric assertional formula in $[Q](\nu)$. With $\mathcal{M}(\Sigma)$ we will indicate the set of all models of an ABox Σ.

Satisfaction of closures is defined on the basis of a notion of minimal knowledge, modelled by epistemic interpretations.

Definition 29 *An* epistemic interpretation *is a pair* $(\mathcal{I}, \mathcal{W})$ *where* $\mathcal{I}$ *is an interpretation and* $\mathcal{W}$ *is a set of interpretations. An epistemic interpretation* satisfies *a closure* $\texttt{Cl}(a)$ *if and only if the following conditions hold:*

[2]Note that γ maps the query parameter ν into Δ.

1. *for every primitive concept symbol* A, $t \in A^{\mathcal{I}}(\gamma(a))$ *iff* $t \in A^{\mathcal{J}}(\gamma(a))$ *for all* $\mathcal{J} \in \mathcal{W}$;
2. *for every primitive concept symbol* A, $f \in A^{\mathcal{I}}(\gamma(a))$ *iff* $t \notin A^{\mathcal{J}}(\gamma(a))$ *for some* $\mathcal{J} \in \mathcal{W}$;
3. *for every primitive role symbol* P *and parameter* $p \in \Delta$, $t \in P^{\mathcal{I}}(\gamma(a), p)$ *iff* $t \in P^{\mathcal{J}}(\gamma(a), p)$ *for all* $\mathcal{J} \in \mathcal{W}$;
4. *for every primitive role symbol* P *and parameter* $p \in \Delta$, $f \in P^{\mathcal{I}}(\gamma(a), p)$ *iff* $t \notin P^{\mathcal{J}}(\gamma(a), p)$ *for some* $\mathcal{J} \in \mathcal{W}$.

An epistemic interpretation satisfies *(is a* model *of) a set of closures if and only if it satisfies each closure in the set.* ∎

Definition 30 *Let* (Σ, Ω) *be a knowledge base. An interpretation* $\mathcal{I}$ satisfies *(is a* model *of)* (Σ, Ω) *if and only if* $\mathcal{I}$ *is a model of* Σ *and* $(\mathcal{I}, \mathcal{M}(\Sigma))$ *is a model of* Ω. ∎

Finally,

Definition 31 *A knowledge base* (Σ, Ω) c-entails *a query* $[Q](\nu)$, *written* $(\Sigma, \Omega) \mid\!\approx_c [Q](\nu)$, *if and only if all models of* (Σ, Ω) *satisfy* $[Q](\nu)$. ∎

In the following, we will use $\mid\!\approx$ as the entailment relation with respect to standard four-valued semantics such that: an ABox Σ *entails* an assertion α, written $\Sigma \mid\!\approx \alpha$, if and only if every standard four-valued interpretation satisfying all assertions in Σ, satisfies also α (see [Straccia, 1995]).

It is easy to verify that, for any model of a knowledge base (Σ, Ω) and closed individual a, $\gamma(a)$ is allowed in the positive extension of a primitive concept A just in case $A(a)$ is entailed by Σ, in symbols $\Sigma \mid\!\approx A(a)$ (similar for roles). In other words, closures force minimal knowledge on closed individuals, ruling out models in which these individuals show up in undue places.

Terminolgical axioms can be included in the model by adopting the devices already mentioned in Chapter 4.

Finally, the IR task can be defined in terms of the retrieval problem. Let (Σ, Ω) be a knowledge base, and the retrieval problem defined as

Retrieval problem: Let $[Q](\nu)$ be a query. What is the set of all the individuals a such that $(\Sigma, \Omega) \mid\!\approx_c [Q](a)$?

As usual, each document is uniquely identified by an individual d, a document base is a set of assertional formulae describing a set of documents, a query is a set of parametric assertional formulae $[Q](\nu)$ (interpreted as disjunction of parametric assertional formulae), *i.e.* possible document descriptions. The IR task is to retrieve all documents d which are "instances" of the query $[Q](\nu)$ and is captured by the retrieval problem defined above.

5.2 Properties of the semantics

This section illustrates the salient features of the logic defined in the previous section; in so doing, we will restrict ourselves to $\mathcal{ALC}$-concepts, for simplicity. Let us consider the following example:

$$\begin{aligned}\Sigma &= \{\texttt{Letter(letter211)}, \texttt{Sender(letter211, umberto)}, \texttt{Italian(umberto)}\}\\ \Omega &= \{\texttt{Cl(letter211)}\}\end{aligned}$$

By virtue of considerations analogous to those developed in Section 3.4, it can be verified that:

$$(\Sigma, \Omega) \not\approx_c \neg\texttt{Book(letter211)}, \text{and}$$
$$(\Sigma, \Omega) \not\approx_c \forall\texttt{Sender.Italian(letter211)}.$$

In particular, in all the models of (Σ, Ω), $\gamma(\texttt{letter211})$ only belongs to the positive extension of `Letter` and, as first member of a pair, to that of `Sender`. As a consequence and on the basis of item 2 of Definition 29, in all the models of (Σ, Ω), $\gamma(\texttt{letter211})$ is not in the positive extension of `Book`, but in the negative extension of `Book`, hence the former entailment relation above. As far as the latter relation, in all the models of (Σ, Ω), $\gamma(\texttt{umberto})$ is the only parameter to be in the positive extension of `Sender` as a second member of a pair whose first element is $\gamma(\texttt{letter211})$; moreover, in all the models of (Σ, Ω), $\gamma(\texttt{umberto})$ is in the positive extension of `Italian`; it follows that, all the models of (Σ, Ω) satisfy $\forall\texttt{Sender.Italian(letter211)}$.

Let C be a concept. We will say that C is *quantifier free* if and only if no $\exists, \forall$ symbols appear in C. Moreover, a knowledge base (Σ, Ω) such that all individuals appearing in the knowledge base are closed, is called *completely closed*. The following propositions directly relate to those presented in Chapter 3 on the properties of the $\models_c$ relation. Their aim is to show the effects of embedding closures in a four-valued semantics.

Proposition 6 *Let (Σ, Ω) be a knowledge base, $\texttt{Cl}(a) \in \Omega$, and $C(a)$ a concept assertion. Then*

1. *either $(\Sigma, \Omega) \not\approx_c C(a)$ or $(\Sigma, \Omega) \not\approx_c \neg C(a)$, for any quantifier free C;*
2. *if (Σ, Ω) is completely closed, then either $(\Sigma, \Omega) \not\approx_c C(a)$ or $(\Sigma, \Omega) \not\approx_c \neg C(a)$, for any C.* ∎

Proposition 7 *Let (Σ, Ω) be a KB and α an assertion $C(a)$ or $R(a, b)$, such that $\mathcal{I}$ is a model of (Σ, Ω). Then the following hold: if $\texttt{Cl}(a) \in \Omega$ then*

1. *if C is quantifier free, then $\mathcal{I}$ satisfies (f-satisfies) α if and only if all models of (Σ, Ω) satisfy (f-satisfy) α;*

2. *if* (Σ, Ω) *is completely closed, then then* $\mathcal{I}$ *satisfies (f-satisfies)* α *if and only if all models of* (Σ, Ω) *satisfy (f-satisfy)* α*, for any* C *and* R. ∎

Corollary 5 *Let* (Σ, Ω) *be a knowledge base,if* $\mathtt{Cl}(a) \in \Omega$ *and* $(C \sqcup D)(a)$ *an assertion. Then:*

1. $(\Sigma, \Omega) \mid\!\approx_c (C \sqcup D)(a)$ *iff* $(\Sigma, \Omega) \mid\!\approx_c C(a)$ *or* $(\Sigma, \Omega) \mid\!\approx_c D(a)$*, for any quantifier free* C *and* D*;*

2. *if* (Σ, Ω) *is completely closed, then* $(\Sigma, \Omega) \mid\!\approx_c (C \sqcup D)(a)$ *iff* $(\Sigma, \Omega) \mid\!\approx_c C(a)$ *or* $(\Sigma, \Omega) \mid\!\approx_c D(a)$*, for any* C *and* D. ∎

We now ask ourselves how c-entailment relates to classical logical entailment, which is denoted as $\mid\!\approx$. The answer to this question is given, as in Chapter 3, in three steps. First, it is shown that a knowledge base with no closures is equivalent to a set of assertions.

Proposition 8 *Let* Σ *be a set of assertions. Then an interpretation* $\mathcal{I}$ *is a model of* $(\Sigma, \emptyset)$ *if and only if* $\mathcal{I}$ *is a model of* Σ. ∎

Second, c-entailment extends classical entailment, that is $\mid\!\approx \subset \mid\!\approx_c$. This relationship is derived in two steps: next Proposition asserts that $\mid\!\approx \subseteq \mid\!\approx_c$.

Proposition 9 *Let* (Σ, Ω) *be a knowledge base and* $C(a)$ *an assertion. Then* $\Sigma \mid\!\approx C(a)$ *implies* $(\Sigma, \Omega) \mid\!\approx_c C(a)$. ∎

In order to show that $\mid\!\approx \neq \mid\!\approx_c$, it suffices to consider the knowledge base (Σ, Ω) defined at the beginning of this Section. As it can been shown, $\Sigma \not\mid\!\approx \neg\mathtt{Book}(\mathtt{letter211})$, whereas $(\Sigma, \Omega) \mid\!\approx_c \neg\mathtt{Book}(\mathtt{letter211})$.

Finally, the next Proposition shows exactly what is the inferential gain of c-entailment over classical entailment.

Proposition 10 *Let* (Σ, Ω) *be a knowledge base and* $\mathtt{Cl}(a) \in \Omega$*. Then for each primitive concept* A*,*

1. $\Sigma \mid\!\approx A(a)$ *implies* $(\Sigma, \Omega) \mid\!\approx_c A(a)$*;*

2. $\Sigma \not\mid\!\approx A(a)$ *implies* $(\Sigma, \Omega) \mid\!\approx_c \neg A(a)$.

Conversely, if (Σ, Ω) *is satisfiable, then for each primitive concept* A*,*

3. $(\Sigma, \Omega) \mid\!\approx_c A(a)$ *implies* $\Sigma \mid\!\approx A(a)$*;*

4. $(\Sigma, \Omega) \mid\!\approx_c \neg A(a)$ *implies* $\Sigma \not\mid\!\approx A(a)$.

The same considerations developed for Proposition 5 apply to the last one, *mutatis mutandis*. In addition, all the above propositions hold for roles, too. For example, we have:

Proposition 11 *Let* (Σ, Ω) *be a knowledge base and* $\mathtt{Cl}(a) \in \Omega$. *Then for each primitive role* P,

1. $\Sigma \mid\!\approx P(a,b)$ *implies* $(\Sigma, \Omega) \mid\!\approx_c P(a,b)$;

2. $\Sigma \not\mid\!\approx P(a,b)$ *implies* $(\Sigma, \Omega) \mid\!\approx_c \neg P(a,b)$.

Conversely, if (Σ, Ω) *is satisfiable, then for each primitive role* P,

3. $(\Sigma, \Omega) \mid\!\approx_c P(a,b)$ *implies* $\Sigma \mid\!\approx P(a,b)$;

4. $(\Sigma, \Omega) \mid\!\approx_c \neg P(a,b)$ *implies* $\Sigma \not\mid\!\approx P(a,b)$. ∎

5.3 A Gentzen style sequent calculus for MIRLOG

In this section we will present a complete Gentzen-like sequent calculus for entailment in MIRLOG which extends the one presented for standard entailment in the previous chapter. The main idea behind our approach is that in order to prove $(\Sigma, \Omega) \mid\!\approx_c [Q](\nu)$, we attempt to prove the sequent $\Sigma \rightarrow_{(\Sigma,\Omega)} [Q](\nu)$, where (Σ, Ω) is an MIRLOG knowledge base and $[Q](\nu)$ is an MIRLOG query. The modularity of this calculus is due to the fact that it is sufficient to develop rules for each operator considered.

First, we recall the retrieval problem, defined as:

Retrieval problem: Let $[Q](\nu)$ be a query. What is the set of all the individuals (the document's identifier) d_{id} such that $(\Sigma, \Omega) \mid\!\approx_c [Q](d_{id})$?

We can further restrict to those Σ and $[Q](d_{id})$ made out of assertional formulae in *Negation Normal Form* (NNF, for short), as shown below.

Definition 32 *A concept* C *is in* Negation Normal Form *iff if* C *contains a concept negation then it is the negation of a primitive concept. An assertion is in* Negation Normal Form *iff the roles and concepts involved are in Negation normal form. An assertional formula* γ *is in* Negation Normal Form *iff if* γ *contains a negation of an assertional formulae* δ, *then* δ *is an assertion in negation normal form.* ∎

Now, first note that there are simple relationships between the operators $\wedge, \vee, \sim$ and $\sqcap, \sqcup, \neg$.

Lemma 10 *Let γ, δ be assertional formulae and let C, D be two concepts and let R be a role. Then the following hold:*

1. $(C \sqcap D)(a) \equiv C(a) \wedge D(a)$;
2. $(C \sqcup D)(a) \equiv C(a) \vee D(a)$;
3. $\sim (\gamma \wedge \delta) \equiv \sim \gamma \vee \sim \delta$;
4. $\sim (\gamma \vee \delta) \equiv \sim \gamma \wedge \sim \delta$;
5. $\sim\sim \gamma \equiv \gamma$;
6. $\sim C(a) \equiv \neg C(a)$;
7. $\sim R(a, b) \equiv \neg R(a, b)$;
8. $\neg\neg C \equiv C$;
9. $\neg\neg R \equiv R$;
10. $\neg(C \sqcap D) \equiv \neg C \sqcup \neg D$;
11. $\neg(C \sqcup D) \equiv \neg C \sqcap \neg D$;
12. $\neg(\forall R.C) \equiv \exists R.\neg C$;
13. $\neg(\exists R.C) \equiv \forall R.\neg C$. ∎

It is easy to see that every assertional formula, assertion and concept can be transformed in polynomial time into an equivalent assertional formula, assertion and concept in NNF, respectively. Note that in MIRLOG roles are already in NNF.

Lemma 11 *Let C be a concept, α be an assertion and let γ be an assertional formula. Then there exists an equivalent concept C', an equivalent assertion α' and an equivalent assertional formula γ' in NNF, respectively, which can be derived in polynomial time.* ∎

In the following we will consider only concepts, assertions and assertional formulae in NNF.

5.3.1 Axiom and rules

Our calculus for entailment in MIRLOG is defined as follows. Assume an alphabet of symbols $\mathcal{V}$ called *variables* (denoted below by x), disjoint from the alphabets introduced so far. The alphabet of *objects*, written $\mathcal{O}^+$, is the union of the alphabets of variables and individuals, *i.e.* $\mathcal{O}^+ = \mathcal{O} \cup \mathcal{V}$, (objects are denoted below by o and o'). An interpretation $\mathcal{I}$ is extended to the objects such that $\mathcal{I}$ maps every variable into Δ and $\mathcal{I}$ is γ on individual constants.

Variables could appear in MIRLOG assertions, hence in assertional formulae.

Definition 33 *Let* (Σ, Ω) *be a knowledge base. A* (Σ, Ω)-sequent*, or simply* sequent*, is an expression of the form* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$*, where* Γ *and* Δ *are respectively sequences* $\gamma_1, \ldots, \gamma_n$ *(*$n \geq 1$*) and* $\delta_1, \ldots, \delta_m$ *(*$m \geq 1$*) of* MIRLOG *assertional formulae. Moreover, each assertional formula of* Σ *must appear in* Γ*.* Γ *is called the* antecedent *and* Δ *is called the* succedent*. Finally, if* $\texttt{Cl}(a) \in \Omega$*, then* $\Gamma \cup \Delta$ *does not contain any expression of the form* $R(a, x)$*.* ∎

In order to simplify notation, we will write $\Gamma \rightarrow \Delta$ in place of $\Gamma \rightarrow_{(\emptyset,\emptyset)} \Delta$.

The intuitive meaning of a sequent $\gamma_1, \ldots, \gamma_n \rightarrow_{(\Sigma,\Omega)} \delta_1, \ldots, \delta_m$ is as follows: let $\mathcal{I}$ be an interpretation such that $(\mathcal{I}, \mathcal{M}(\Sigma))$ satisfies Ω. If $\mathcal{I}$ satisfies $\{\gamma_1, \ldots, \gamma_n\}$ then I satisfies the query $[\{\delta_1, \ldots, \delta_m\}](\nu)$, *i.e.* satisfies some δ_j. Our aim is to show that: $\Sigma \rightarrow_{(\Sigma,\Omega)} [Q](\nu)$ is provable from the axioms and the rules of our calculus if and only if $(\Sigma, \Omega) \mid\!\!\approx_c [Q](\nu)$.

It should be noted that the semantics of sequents suggests that instead of using sequences of assertional formulae, we could have used sets, *e.g.* it happens that $\Sigma \subseteq \Gamma$. We could indeed define sequents in terms of finite *sets* Γ, Δ of assertional formulae. For simplicity we will use the same notation for both.

As first, let $\mathcal{I}$ be an interpretation, let (Σ, Ω) be a KB and let T be a set of assertional formulae. We will say that $\mathcal{I}$ (Σ, Ω)-*satisfies* T if and only if $\mathcal{I}$ satisfies T and $(\mathcal{I}, \mathcal{M}(\Sigma))$ satisfies Ω. Thus, $\mathcal{I}$ (Σ, Ω)-*satisfies* Σ if and only if $\mathcal{I}$ is a model of (Σ, Ω).

A sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is *satisfiable* if and only if there is an interpretations $\mathcal{I}$ such that if $\mathcal{I}$ (Σ, Ω)-satisfies Γ then $\mathcal{I}$ satisfies some $\delta_j \in \Delta$.

A sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is *valid* iff all interpretations satisfy $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$. A sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is called *falsifiable* iff it is not valid.

For example, the sequent $(C \sqcap D)(a) \rightarrow C(a)$ is valid, whereas the sequent $(C \sqcup D)(a) \rightarrow C(a)$ is not (*i.e.* it is falsifiable). From the above definition it follows easily that $(\Sigma, \Omega) \mid\!\!\approx_c [Q](\nu)$ if and only if $\Sigma \rightarrow_{(\Sigma,\Omega)} [Q](\nu)$ is valid.

The rules operating on sequents fall naturally into two categories: those operating on assertional formulae occurring in the antecedent, and those on assertional formulae occurring in the succedent. The application of a rule may cause a sequent to be split into two sequents.

Definition 34 (Axioms) *Let* (Σ, Ω) *be a knowledge base.* Axioms *are sequents of the form*

1. $\alpha, \Gamma \rightarrow_{(\Sigma,\Omega)} \alpha, \Delta$;
2. $\Gamma \rightarrow_{(\Sigma,\Omega)} \top(o), \Delta$;
3. $\bot(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$;
4. $\Gamma \rightarrow_{(\Sigma,\Omega)} \neg A(a), \Delta$ *if* $\mathtt{Cl}(a) \in \Omega$ *and* $\Sigma \not\approx A(a)$,
5. $\Gamma \rightarrow_{(\Sigma,\Omega)} \neg P(a,b), \Delta$ *if* $\mathtt{Cl}(a) \in \Omega$ *and* $\Sigma \not\approx P(a,b)$,

where α *is a primitive or negated primitive assertion*[3]*,* a, b *are individuals and* o *is an object.* ∎

Every rule consists of one or two upper sequents called *premisses* and of a lower sequent called *conclusion*. The rules of our calculus are then defined on NNF expressions as follows[4].

Definition 35 (Rules) *Let* (Σ, Ω) *be a knowledge base. The* inference rules *of the sequent calculus for* MIRLOG *are the following:*

rule $(\wedge \rightarrow)$**:**

$$(\wedge\rightarrow)\ \frac{\gamma, \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{\gamma \wedge \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

rule $(\rightarrow \wedge)$**:**

$$(\rightarrow\wedge)\ \frac{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, \gamma \quad \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, \delta}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, \gamma \wedge \delta}$$

rule $(\vee \rightarrow)$**:**

$$(\vee\rightarrow)\ \frac{\gamma, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta \quad \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{\gamma \vee \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

rule $(\rightarrow \vee)$**:**

$$(\rightarrow\vee)\ \frac{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, \gamma, \delta}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, \gamma \vee \delta}$$

rule $(\bot_{C1}\rightarrow)$**:** *if* $\mathtt{Cl}(a) \in \Omega$ *and* $\Sigma \not\approx A(a)$ *then*

$$(\bot_{C1}\rightarrow)\ \frac{\bot(a), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{A(a), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

[3]Note that α could be a generic assertion or assertional formula. The restriction on α is used only for developing easier proofs.

[4]Note that we are not looking for optimization of the rules.

rule $(\bot_{C2}\rightarrow)$**:** *if* $\mathtt{Cl}(a) \in \Omega$ *then*

$$(\bot_{C2}\rightarrow)\ \frac{\bot\,(a), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{A(a), \neg A(a), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

rule $(\bot_{R1}\rightarrow)$**:** *if* $\mathtt{Cl}(a) \in \Omega$ *and* $\Sigma \not\approx P(a,b)$ *then*

$$(\bot_{R1}\rightarrow)\ \frac{\bot\,(a), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{P(a,b), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

rule $(\bot_{R2}\rightarrow)$**:** *if* $\mathtt{Cl}(a) \in \Omega$ *then*

$$(\bot_{R2}\rightarrow)\ \frac{\bot\,(a), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{P(a,b), \neg P(a,b), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

rule $(\sqcap \rightarrow)$**:**

$$(\sqcap \rightarrow)\ \frac{C(o), D(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{(C \sqcap D)(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

rule $(\rightarrow \sqcap)$**:**

$$(\rightarrow \sqcap)\ \frac{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, C(o) \quad \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, D(o)}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, (C \sqcap D)(o)}$$

rule $(\sqcup \rightarrow)$**:**

$$(\sqcup \rightarrow)\ \frac{C(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta \quad D(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{(C \sqcup D)(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

rule $(\rightarrow \sqcup)$**:**

$$(\rightarrow \sqcup)\ \frac{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, C(t), D(o)}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, (C \sqcup D)(o)}$$

rule $(\forall \rightarrow)$**:**

$$(\forall \rightarrow)\ \frac{(\forall R.C)(t), R(o,o'), C(o'), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{(\forall R.C)(t), R(o,o'), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

rule $(\rightarrow \forall)$**:**

$$(\rightarrow \forall)\ \frac{\Gamma', \rightarrow_{(\Sigma,\Omega)} \Delta, \delta}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, (\forall R.C)(o)}$$

1. *if* $\mathtt{Cl}(o) \notin \Omega$ *then* Γ' *is* $\Gamma \cup \{R(o,x)\}$ *and* δ *is* $C(x)$*;*
2. *if* $\mathtt{Cl}(o) \in \Omega$ *then* δ *is given by:*
   - $\delta = C(a_1) \wedge \ldots \wedge C(a_n)$ *where* $a_i \in \mathcal{O}$*, with* $(i > 0)$*, are all the individuals such that:* $\Sigma \not\approx P(o,a_i)$*if* $R = P$*, else* $\Sigma \not\approx P(o,a_i)$*;*
   - $\delta = \top(o)$ *otherwise;*

and Γ' *is given by:*

- $\Gamma' = \Gamma \cup \{R(o,a_1), \ldots R(o,a_n)\}$ *where* $a_i \in \mathcal{O}$, *with* $(i > 0)$, *are all the individuals such that:* $\Sigma \mid\!\approx P(o,a_i)$ *if* $R = P$, *else* $\Sigma \not\mid\!\approx P(o,a_i)$;
- $\Gamma' = \Gamma$ *otherwise.*

rule $(\exists \rightarrow)$**:**

$$(\exists \rightarrow) \quad \frac{\gamma, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}{(\exists R.C)(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

1. *if* $\mathtt{Cl}(o) \notin \Omega$ *then* γ *is* $\{R(o,x), C(x)\}$;
2. *if* $\mathtt{Cl}(o) \in \Omega$ *then* γ *is given by:*
   - $\gamma = (R(o,a_1) \wedge C(a_1)) \vee \ldots \vee (R(o,a_n) \wedge C(a_n))$ *where* $a_i \in \mathcal{O}$, *with* $(i > 0)$, *are all the individuals such that:* $\Sigma \mid\!\approx P(o,a_i)$ *if* $R = P$, *else* $\Sigma \not\mid\!\approx P(o,a_i)$;
   - $\gamma = \bot\,(o)$ *otherwise.*

rule $(\rightarrow \exists)$**:**

$$(\rightarrow \exists) \quad \frac{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, (\exists R.C)(o), R(o,o') \wedge C(o')}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, (\exists R.C)(o)}$$

where x *is a new variable (called also* eigenvariable*) which does not appear in the conclusion of the rules and* o, o' *are objects.* ∎

5.3.2 Provability, Soundness and Completeness

Every inference rule can be represented as a tree with two nodes if the rule has a single premise, or three nodes if the rule has two premises. In both cases, the root of the tree is labeled with the conclusion of the rule and the leaves (called sons) are labeled with the premises.

Definition 36 *A* deduction tree *is a tree whose nodes are labeled with a sequent, and is closed under the rules of Definition 35 in the following sense:*

1. *every node labeled with a sequent is a deduction tree;*
2. *for any deduction tree* T' *whose root is labeled* $\Gamma' \rightarrow_{(\Sigma,\Omega)} \Delta'$, *for any instance of a rule with premise* $\Gamma' \rightarrow_{(\Sigma,\Omega)} \Delta'$ *and conclusion* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$, *the tree* T *whose root is labeled with* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *and has as unique son the root of* T' *is a deduction tree;*
3. *for any two deduction trees* T' *and* T'' *whose root are labeled* $\Gamma' \rightarrow_{(\Sigma,\Omega)} \Delta'$, *and* $\Gamma'' \rightarrow_{(\Sigma,\Omega)} \Delta''$, *respectively, for any instance of a rule with premises* $\Gamma' \rightarrow_{(\Sigma,\Omega)} \Delta'$ *and* $\Gamma'' \rightarrow_{(\Sigma,\Omega)} \Delta''$ *and conclusion* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$, *the tree* T *whose root is labeled with* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *and has as sons only the roots of* T' *and* T'', *is a deduction tree.*

The sequent labeling the root of a deduction tree is called conclusion *of the deduction tree. The* depth *of a deduction tree is defined as the length of the longest path from the root to the leaves.* ∎

Definition 37 *A* proof tree *is a deduction tree whose leaves are labeled with an axiom.* ∎

Definition 38 *A* counterexample tree *is a deduction tree such that some leaf is labeled with a falsifiable sequent.* ∎

Definition 39 *A sequent* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is* provable, *written* $\mid\!\sim_c \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$, *iff there is a proof tree of which it is the conclusion.* ∎

We are going now to prove the soundness of our calculus: *i.e.* if the sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is provable then $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid. The following lemma is easily verified.

Lemma 12 *No axiom is falsifiable. Equivalently, every axiom is valid.* ∎

The soundness of the rules is proven by the following lemma.

Lemma 13 *For each of the rules in Definition 35, the conclusion of a rule is falsifiable iff at least one of the premises of the rule is falsifiable. Equivalently, the conclusion of a rule is valid iff all premises of the rule are valid.* ∎

Using the two lemmas above, we are ready to prove the soundness of our calculus.

Theorem 6 (Soundness) *If a sequent* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is provable, then it is valid.* ∎

A simple corollary is the soundness of our calculus wrt entailment.

Corollary 6 *Let* (Σ, Ω) *be a knowledge base and* $[Q](\nu)$ *a query. Then* $\mid\!\sim_c \Sigma \rightarrow_{(\Sigma,\Omega)} [Q](\nu)$ *implies* $(\Sigma, \Omega) \mid\!\approx_c [Q](\nu)$. ∎

We are going now to prove the completeness of our calculus: *i.e.* if the sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid then $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is provable.

We begin with signed assertional formulae. Let T and F be two new symbols not appearing in the considered alphabets, and consider the *extended language wrt* Δ which is obtained by adding to the set of individuals a set

$$\{\mathbf{p} \mid p \in \Delta\}$$

of new constants, one for each element of Δ. γ is extended to the constants in the set $\{\mathbf{p} \mid p \in \Delta\}$ by defining

$$\gamma(\mathbf{p}) = p$$

Definition 40 Signed assertional formulae of type a, type b, type c *and* type d *and their* components *are defined as follows (C, D are concepts, R is a role, o, o' are objects, and γ, δ are assertional formulae):*

- type-a signed assertional formulae:

γ^a	γ_1^a	γ_2^a
$T((C \sqcap D)(o))$	$T(C(o))$	$T(D(o))$
$NT((C \sqcup D)(o))$	$NT(C(o))$	$NT(D(o))$
$T(\gamma \wedge \delta)$	$T(\gamma)$	$T(\delta)$
$NT(\gamma \vee \delta)$	$NT(\gamma)$	$NT(\delta)$

- type-b signed assertional formulae:

γ^b	γ_1^b	γ_2^b
$T((C \sqcup D)(o))$	$T(C(o))$	$T(D(o))$
$NT((C \sqcap D)(o))$	$NT(C(o))$	$NT(D(o))$
$T(\gamma \vee \delta)$	$T(\gamma)$	$T(\delta)$
$NT(\gamma \wedge \delta)$	$NT(\gamma)$	$NT(\delta)$

- type-c signed assertional formulae:

γ^c	γ_1^c	γ_2^c
$T((\forall R.C)(o))$	$T(R(o, o'))$	$T(C(o'))$
$NT((\exists R.C)(o))$	$T(R(o, o'))$	$NT(C(o'))$

- type-d signed assertional formulae:

γ^d	γ_1^d	γ_2^d
$T((\exists R.C)(o))$	$T(R(o, o'))$	$T(C(o'))$
$NT((\forall R.C)(o))$	$T(R(o, o'))$	$NT(C(o'))$

∎

Let γ be an assertional formula. Then $T(\gamma)$ and $NT(\gamma)$ are called *conjugated signed assertional formulae.*

We define then satisfaction of signed assertional formulae as follows. Let $\mathcal{I}$ be an interpretation and γ an assertional formula. Then $\mathcal{I}$ satisfies $T(\gamma)$ iff $\mathcal{I}$ satisfies γ, whereas $\mathcal{I}$ satisfies $NT(\gamma)$ iff $\mathcal{I}$ does not satisfy γ.

The following lemma holds.

Lemma 14 *Let $\mathcal{I}$ be an interpretation. Then:*

1. *for any signed assertional formula γ^a of type a, $\mathcal{I}$ satisfies γ^a iff $\mathcal{I}$ satisfies both γ_1^a and γ_2^a;*

2. *for any signed assertion γ^b of type b, $\mathcal{I}$ satisfies γ^b iff $\mathcal{I}$ satisfies γ_1^b or γ_2^b;*

3. *for any signed assertion γ^c of type c, $\mathcal{I}$ satisfies γ^c iff if $\mathcal{I}$ satisfies γ_1^c then $\mathcal{I}$ satisfies γ_2^c, for every* **p** *in place of o';*

4. *for any signed assertion γ^d of type d, $\mathcal{I}$ satisfies γ^d iff $\mathcal{I}$ satisfies γ_1^d and γ_2^d, for at least one* **p** *in place of o'.* ∎

Signed assertional formulae of type a are also called assertional formulae of *conjunctive type*, signed assertional formulae of type b are called assertional formulae of *disjunctive type*, signed assertional formulae of type c are called assertional formulae of *universal type* and signed assertional formulae of type d are called assertional formulae of *existential type*.

In order to prove the completeness we will use Hintikka sets.

Definition 41 *Let H be a set of objects. A* Hintikka set S wrt H *is a set of signed assertional formulae such that the following conditions hold for all signed assertional formulae γ^a, γ^b, γ^c, γ^d of type a,b,c,d, respectively:*

1. *H0: No conjugated signed primitive assertions or conjugated signed negated primitive assertions are in S. Moreover, neither $T(\bot(a))$ nor $NT(\top(a))$ are in S;*

2. *H1: If a type-a assertional formula γ^a is in S, then both γ_1^a and γ_2^a are in S;*

3. *H2: If a type-b assertional formula γ^b is in S, then either γ_1^b is in S or γ_2^b is in S;*

4. *H3: If a type-c assertional formula γ^c is in S, then for every object $o' \in H$, if γ_1^c is in S then γ_2^c is in S;*

5. *H4: If a type-d assertional formula γ^d is in S, then there is at least one object $o' \in H$ such that both γ_1^d and γ_2^d are in S.* ∎

Lemma 15 *Every Hintikka set S wrt a set of objects H is satisfiable.* ∎

In order to prove the completeness, our goal is, as in Chapter 4, to attempt to falsify the given sequent. To this end, the procedure **Search** presented in Section 4.4.3 can be applied, with the only care of modifying its auxiliary procedures GrowLeft and GrowRight so to have them handling all the rules of the present calculus, as they are specified above.

Lemma 16 *The* **Search** *procedure satisfies the following conditions:*

1. *If the input sequent* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is valid, then the procedure* **Search** *halts with a finite closed tree* T *which is a proof tree for* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

2. *If the input sequent* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is falsifiable, either* **Search** *halts with a finite counterexample tree* T *and* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is falsifiable, or* **Search** *generates an infinite tree* T *and* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is falsifiable.* ∎

As a consequence we obtain the completeness theorem.

Theorem 7 (Completeness) *If a sequent* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is valid, then it is provable.* ∎

Corollary 7 *Let* (Σ, Ω) *be a knowledge base and* $[Q](\nu)$ *a query. Then* $\mid\sim_c \Sigma \rightarrow_{(\Sigma,\Omega)} [Q](\nu)$ *if and only if* $(\Sigma, \Omega) \mid\approx_c [Q](\nu)$. ∎

5.4 Craig's "relevant" interpolation theorem

In this section we will extend the material presented in Section 4.5 to the present framework.

Definition 42 *Let* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *be a valid sequent. Then an* interpolant *of* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is an assertional formula* γ *such that*

1. $\Gamma \rightarrow_{(\Sigma,\Omega)} \gamma$ *is a valid sequent formula;*

2. $\gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is a valid sequent formula;*

3. *every primitive concept* A *occurring in* γ,

   (a) *occurs both in* Γ *and* Δ, *or*

   (b) A *occurs both in* γ *and* Δ *in the form* $\neg A(a)$, *if* $\mathtt{Cl}(a) \in \Omega$ *and* $\Sigma \not\mid\approx A(a)$;

4. *every primitive role* P *occurring in* γ,

   (a) *occurs both in* Γ *and* Δ, *or*

   (b) P *occurs both in* γ *and* Δ *in the form* $\neg P(a, b)$, *if* $\mathtt{Cl}(a) \in \Omega$ *and* $\Sigma \not\mid\approx P(a, b)$;

5. *neither* $\bot$ *nor* $\top$ *appear in* γ. ∎

Now we are able to formalize the main theorem.

Theorem 8 (Four-valued interpolation theorem) $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is a valid sequent iff one of the following cases hold:*

1. *there exists a constructible interpolant* γ *of* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$*;*

2. Γ *is not* (Σ, Ω)*-satisfiable;*

3. $(\bigvee_{\delta \in \Delta}) \equiv \top(a)$*, for some* a*.* ∎

Inspecting the structure of γ, γ can be seen as the part of relevant information which is in (Σ, Ω) in order to answer the query $[Q](\nu)$, *i.e.*the interpolant can be seen as an "explanation" of the query. That is, in order $(\Sigma, \Omega) \mid\!\approx_c [Q](\nu)$ to hold, the structural components of $[Q](\nu)$ must have an analogue in Σ, modulo MPR, or $[Q](\nu)$ contains the negation of primitive concepts and roles not entailed (by means of $\mid\!\approx$) by Σ.

Also relevant to IR, is the fact that interpolants can be used for user relevance feedback purposes. The basic idea is to let the user classify the documents returned by a query into one of the categories: "relevant", "non-relevant" and "I-don't-know". The system uses this information to generate a new query, constructued by using the interpolants of the documents calssified as relevant. The result of the new query is guaranteed to include the documents classified as relevant *and* not to include the documents classified as non-relevant. On the I-don't-know documents, the query acts as a filter, retaining only those that bear some similarity with the relevant ones. The mechanism can be extended to consider as I-don't-know documents *all* the documents of the document base, other than the relavent and the non-relevant ones, to be I-don't-know documents. Our state of knowledge on both these mechanisms is still incomplete. We plan to investigate the effect of the described filtering device in the next future, taking into account the considerations on relevance dynamics presented in the next Chapter of the present document.

Let (Σ, Ω) be the document base and $[Q](\nu)$ be a query. Let $AS((\Sigma, \Omega), [Q](\nu))$ be the *answer Set* wrt (Σ, Ω) and $[Q](\nu)$, defined as:

$$\{d : (\Sigma, \Omega) \mid\!\approx_c [Q](d)\}$$

and γ_d be an interpolant of $[Q](d)$. Suppose now that the user selects two disjoint subsets of $AS((\Sigma, \Omega), [Q](\nu))$, S_{URFS}, the *User Relevant Feedback Set,* and S_{UIRFS}, the *User IrRelevant Feedback Set,* as those documents relevant and non-relevant to her purposes, respectively. Given (S_{URFS}, S_{UIRFS}), a new query, called *filtering query*, can be constructed by means of the *User Feedback Representation* function, f_{UFR}, which takes into account the user's relevance judgement.

Let γ be an assertional formula and let $\gamma[o/o']$ be the assertional formula obtained from γ by substituting object o with o'. Let $[\gamma_R](\nu)$ and $[\gamma_{IR}](\nu)$ defined as follows:

$$\begin{aligned} [\gamma_R](\nu) &= \bigvee_{d \in S_{URFS}} \gamma_d[d/\nu] \\ [\gamma_{IR}](\nu) &= \bigvee_{d \in S_{UIRFS}} \gamma_d[d/\nu] \end{aligned}$$

$[\gamma_R](\nu)$ is intended to represent the user's relevance, whereas $[\gamma_{IR}](\nu)$ purports at modelling the user's irrelevance feedback. If we would like that the new answer set includes the documents identified by S_{URFS} and will not include documents identified by S_{UIRFS}, we must check if (S_{URFS}, S_{UIRFS}) is *safe*, *i.e.* that there is no $d \in S_{URFS} \cup S_{UIRFS}$ such that

$$(\Sigma, \Omega) \not\approx_c [\gamma_R](d) \wedge \sim [\gamma_{IR}](d)$$

This last condition ensures that there is no document involved in the user's feedback which are both relevant and not relevant.

Now, the filtering query can be defined as:

$$f_{UFR}(\nu) = [Q](d) \wedge [\gamma_R](\nu) \wedge \sim [\gamma_{IR}](\nu)$$

which acts as a filter to the set $AS((\Sigma, \Omega), [Q](\nu))$, in the sense that its answer set will be a subset of $AS((\Sigma, \Omega), [Q](\nu))$.

Note that $f_{UFR}(d)$ is such that a document d is retrieved from a document base if and only if d satisfies some of the user's relevant feedback information S_{URFS} (condition $\bigvee_{d \in S_{URFS}} \gamma_d$) and all of the user's irrelevant feedback information S_{UIRFS} (condition $(\bigwedge_{d \in S_{URFS}} \sim \gamma_d)$).

If we want the filter to apply to the whole document base, we must adopt the following filtering query:

$$f_{UFR}(\nu) = [\gamma_R](\nu) \wedge \sim [\gamma_{IR}](\nu)$$

Of course, the full impact of Theorem 8 to relevance feedback purposes needs further investigations.

Chapter 6

Conclusions

We have presented MIRLOG, a four-valued terminological logic for information retrieval, allowing closure assertions and complex assertional formulae. The logic builds on existing results in the area of knowledge representation; in particular, the syntax and semantics of terminological logics have been extensively used in the definition of MIRLOG, as well as some previous work on relevance terminological logics. However, despite the fact that it has been conceived in an information retrieval setting, MIRLOG makes, to the best of our knowledge, several contributions to the research on logical formalisms for conceptual modelling:

- closure assertions represent an original approach to the problem of endowing a logic with a fully controlled form of closed world reasoning;
- the particular four-valued semantics adopted for MIRLOG is novel and overcomes inferential limitations of previous approaches;
- the calculus for performing retrieval of MIRLOG documents is new, and seems a very promising direction for unifying reasoning on terminological logic both with a two- and a four-valued semantics.

The whole model is an original contribution to logic-based information retrieval modelling, with a special mention for the idea of using Craig's interpolants for modelling relevance feedback.

The next points on the MIRLOG agenda are now: evaluation and extension.

Evaluation will take place both at the theoretical and practical level. Theoretically, we plan to further investigate the properties of the model, in the vein of the work presented in Section 5.2, and also at the computational level, by trying to classify the complexity of the retrieval problem. Practically, we plan to develop a prototypical retrieval engine implementing the calculus given in Section 5.3. The latter activity is part of the FERMI agenda, as well as is the experimentation of the obtained model on a realistic application.

As far as extension is concerned, we are well aware of the fact that the model described in this document is only the logic kernel of a realistic information retrieval system. Two importants directions of extension are foreseen in the context of FERMI in the next future. First, the inclusion in MIRLOG of uncertainty. Second, the development, on top of MIRLOG, of a multimedia information retrieval model. Needless to say, both these extensions will bring MIRLOG much closer to the reality of present and future information retrieval systems. Nevertheless, there are several interesting issues that need be addressed beyond those specifically identified by FERMI.

One of the advantages of the terminological model of information retrieval is that it can profit of the many research results obtained in the field of terminological logics. In Appendix A we review the many extensions to terminological logics that may be of interest to the present context.

Part II

A Study Of System And User Relevance In Information Retrieval

Chapter 7

A Study Of System And User Relevance In Information Retrieval

Information Retrieval systems are based on a notion of relevance of documents for a given user's need expressed by a query in a given formalism. In this document, we propose to formalize some basic IR notions. This formalization must be a kind of specification of what should be an IR system from our point of view. We propose in the last section a partial instantiation of our approach in a modal logic. This paper ends by an example that illustrates the use of this logic.

7.1 Introduction

The scope of IR (Information Retrieval) is to design systems able to provide information to users who are expressing their information needs using query languages in the context of suitable man-machine interactions. Answers are in the form of documents that suit the users' needs. This is the classical notion of relevance [Cooper, 1971] related to the usefulness of a document for a user.

An IRS (Information Retrieval System) is a tool that helps the user in selecting documents as *supports* of information, and not the *needed* information itself. The human user is the only ultimate judge about the relevance of what is retrieved by the system, compared to his needs. By this, we mean that the satisfaction or un-satisfaction of the user at the end of a retrieval session is *the only fact* we must take into consideration. Hence, the main problem of an IRS is approximating this notion of relevance by imitation. System computes relevance measures using data extracted from documents that are called index, using user's need expression that is called query and using a matching function between index and query that retrieves and ranks documents.

The objective of task T13 "Modeling the relevance of retrieval" is to investigate sev-

eral possible ways for implementing the main underlying idea of logic-based information retrieval: a real document d is relevant to a real need (a query) q iff d somehow implies q. This implies first to find, among the notions of implication ("$\rightarrow$") which are formalized by different classes of logics, the one that is most suitable for modeling the relation of relevance of a document d to a query q in terms of the validity of the formula $D \rightarrow Q$, where D and Q are respectively *models* of real objects d and q.

Classical logics are obviously inadequate (one would say: only partially adequate) to cope with this problem. Some main reasons for that may be listed: the necessity to integrate an uncertain evaluation of this implication, the necessity to cope with partial models of documents and queries, the fact that the material implication is obviously too limited. So we are deemed to investigate a number of "non classical" logics that express more closely the "natural" behavior of the relevance relationship.

At this phase of the project, and before opting for the study of a given logic, it seems useful to investigate some basic aspects which are of prime importance when considering IR and Logics.

In this paper, we investigate the underlying concepts hidden behind the IR notion of relevance and we propose for that to formalize these notions. We analyze in more details why we think that this notion of relevance is related to logic in a natural way, and we explain why, in our opinion, relevance is inherent to the notion of modality. This will in turn lead us to investigate the possible use of modal logic to express the relation of relevance.

This approach will raise the problem of defining what is relevance, what is a query, and what is considered as information in documents. We present in the following some basic facts, hypotheses and definitions that we will use to ground our modal model for IR.

7.1.1 Information and document

If we consider any real document d as a signal that supports a certain amount of *information*, the user of an IRS can be viewed as the *receptor* of this signal. He is able to decrypt the signal using his *knowledge* to extract from it some useful information. That information can then be transformed into some new knowledge, if it is relevant to his needs.

When the user asks for information, he submits a query to the IRS. This query usually expresses information that is *related*, or *about*, what he is searching for. Queries q are supposed to reflect the topic the user is interested in. The role of the IRS is to compare the *representation* Q of the query q to some information D extracted from documents d. We call this information D an "index", because it mainly serves to select documents. Since the users' needs are more focused on global topics of documents, than on detailed contents. So an IR index is mainly meant to capture documents' topics.

Classical systems are keyword-based and therefore automated extraction of keywords

can be considered as a very raw and unstructured description of the information contained in the signal associated to a document. We are convinced that the only way of improving an IRS is to extract more information from the signal, and to give it an adequate structure. The key point then is the amount of information we need and in which directions (or facets) we must extract these information, to improve results. In that context, the information facets are *points of view* taken by the receptor of the signal (i.e, the men which read the document). Depending on the way the reader looks at a document and depending on his knowledge, he will read different information. This fact is even more obvious when one considers pictures or videos as documents. We want to consider a general IR model that includes this user behavior.

Before going further on logical model for IR, we want to investigate more closely the notion of relevance. We will propose some basic hypothesis used as postulates when designing new models and derived new IR systems.

7.1.2 The IR notion of relevance

All IR systems offer a matching engine that selects an amount of documents related to a query. All these systems hence are based on the implicit assumption of relevance and try to implement it through matching functions. The notion of relevance is not obvious and it is often compared to the notion of topic relevance [Green, 1995a; Green, 1995b], it is also presented as an aboutness relation [Marron, 1977] or as a utility relation [Cooper, 1971]. Saracevic in [Saracevic, 1976] concluded that various definitions of relevance fell into a general pattern that expresses different notion of relevances : a A of a B existing between a C and a D as determined by E. Slots can be filled by :

A measure, degree, dimension, estimate, appraisal, relation;

B correspondence, utility, connection, satisfaction, fit, bearing, matching;

C document, article, textual form, reference, information provided, fact;

D query, request, information used, use of information, point of view, information requirement statement;

E person, judge, user, requester, information specialist.

He raises different aspects of the relevance notion: one of it is the notion of judgment of a given person following a particular point of view. He also mixes the notion of user's relevance and system's relevance. One of our goals in this work, is to provide an underlying basis for the study of relevance by defining hypothesis and definitions of relevance in order to explore different interpretations of this notion, and so to define guidelines for the design of IRS. We use the notion of user's relevance versus system's relevance and we introduce the notion of point of view on the user and system side within the facet notion. We also deals with the notion of a retrieval session by introducing time in our modelization.

7.2 User relevance

We make a distinction between the notion of *abstract relevance judgment* that is used in test collections and *user's relevance judgment* or *actual relevance judgment* that a user provides when he is consulting some documents retrieved by a system. Relevance judgment of a user must be understood as a reaction of this person after reading a document. For example, this situation occurs when a person is looking for a document in a real library: he consults manual indexes, moves around, takes some books, reads quickly some of them, and in the good case keeps a few books that are about what he is searching for.

In an IRS, the system is supposed to guide the user through a sort of virtual library but with an initial need expressed in some language. We could imagine systems without any initial query at all whose role would be only to present books to the user and helps him like a human library assistant. This is not the case at the moment and the system has to start from an initial definition of the users' needs. Then it proposes a selection of documents to users who partially consult them and express in return what they think about this document selection. The users have to estimate the outputs from the system which in turn takes into account his agreements or disagreements about the retrieved documents. This is called user relevant feedback. In the following paragraphs, we present requirements needed to formalize the notion of relevance and the way it is used in an IR model to build a system.

7.2.1 Basic hypothesis

Let d be a real document and q be the initial user need. We suppose users to be able to declare that a document is relevant or not, or to declare that he does not know about the relevance of a document. This hypothesis is both fundamental and debatable. It is fundamental because if we try to modelize this relevance notion then we have of course to suppose its existence. It is debatable because in practice, relevance is a subjective notion in the sense that it depends on the subject (i.e. the user) which gives it a value. It depends on the user's knowledge of d, on his own general knowledge, and on the precision with which he defines his own need q. Moreover, one cannot guarantee that his need will be both precise and stable in time within a retrieval session.

Hypothesis 1 (Existence of user relevance) *We admit that there could exist a relationship between d and q that the user call "relevance". This notion is not absolute but relative to a given user.*

In the same way we propose to examine the fact that a user can know and express that a given document is not relevant to his needs.

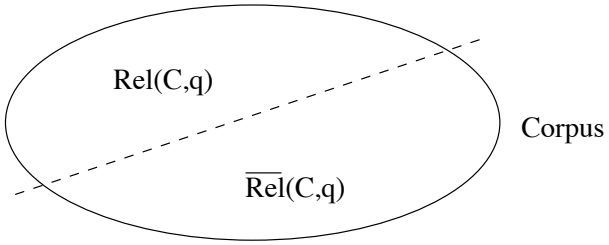

Figure 7.1: Abstract user relevance

Hypothesis 2 (Existence of user irrelevance) *We admit that there could exist a relationship between d and q that the user call "irrelevance", this notion is relative to a given user.*

Definition 1 (Abstract user relevance) *If a user could know all the documents of the corpus, he would be able to decide whether a document is relevant or not to his information needs. Then we obtain an "ideal" partition of the corpus that we would call* **abstract user relevance**.

Let $\mathcal{C}$ be the corpus of documents, $Rel(\mathcal{C}, q)$ the documents relevant in an abstract way and $\overline{Rel}(\mathcal{C}, q)$ the document irrelevant also in an abstract way. Then we have :

$$\mathcal{C} = Rel(\mathcal{C}, q) \cup \overline{Rel}(\mathcal{C}, q)$$

$$Rel(\mathcal{C}, q) \cap \overline{Rel}(\mathcal{C}, q) = \emptyset$$

In actual situations, a user cannot always decide whether a document he is looking at satisfies his needs or not. Sometimes, he cannot decide, because in practice he would have to read the document in more details to have a correct opinion about this document. All in all, his decision takes place only for documents he his able to look at. That's why we distinguish **abstract and actual user relevance**. In the next part, we discuss this notion of relevance.

7.2.2 Characteristics of actual user relevance

We introduce now this notion of actual user relevance with a notation.

Definition 2 (Actual user relevance) *The user relevance or irrelevance judgment about retrieved documents obtained from his need q is called* **actual user relevance**, *and is noted* $\mathcal{R}_q(u)$ *for relevance judgment and* $\overline{\mathcal{R}_q}(u)$ *for irrelevance judgment.*

An important aspect we must take into account is the *time* during which the retrieval session occurs. The users' decision about relevance may change along the session, and also his information need may evolve in the same time. All these changes are triggered by the interactive process of the session during which the user consults proposed documents and input relevance judgments to the system. In our opinion, the well-known process of relevance feedback would thus have to be revisited in the context of a time dependant interaction.

As a consequence, we propose to stamp relevance relation with time value. At the beginning of a retrieval session, the user may not be able to assert the relevance of some document because the system had not show them yet. So the actual relevance relation must be three valued: user knows that a document is relevant, or he knows that a document is not relevant, and finally he could have no opinion about this relevance.

Hypothesis 3 (Actual relevance and time) *Actual relevance is time related and is noted* $\mathcal{R}_q(t,u)$ *and represent the set of relevant documents selected by a user* u *at time* t *for a need* q*. Actual irrelevance is noted* $\overline{\mathcal{R}_q}(t,u)$*. The time* t *is a discrete integer and correspond to the time when user new input is taken into account by the system for a new computation.*

For example, time $t = 0$ corresponds to the time where the initial query is submitted to the system whereas time $t = 1$ corresponds to the time when the system has shown the result of its first computation and the user has express his first relevance feedback. Notice that these time values do not reflect real time but only a chronology of events. At the beginning of the session (at $t = 0$) the sets $\mathcal{R}_q(t,u)$ and $\overline{\mathcal{R}_q}(t,u)$ are empty.

Actual user relevance is the set of all documents judged as relevant during all the session.

$$\mathcal{R}_q(u) = \bigcup_t \mathcal{R}_q(t,u)$$
$$\overline{\mathcal{R}_q}(u) = \bigcup_t \overline{\mathcal{R}_q}(t,u)$$

We also think that there could exist at the same time different points of view from which a user can estimate the relevance value of the system's answer. These points of view are related to the document itself and correspond to different visions of a document. For example, one can consider shapes, colors, emotional content, (etc) of an image.

We call *facets* these different elements found in documents. In multimedia documents, facets are easily understandable. For example, in a picture there exists the physical facet that deals with physical aspect like color, or light; there exists also the geometrical facet that deals with lines, surfaces, etc.

Hypothesis 4 (Actual relevance and facet) *Actual relevance is facet related and is noted* $\mathcal{R}_q(t,f,u)$*. This represents the set of relevant documents selected by a user* u *at time* t *for a neep* q *and according to a particular facet* f*.*

7.2.3 Time dependant properties of actual relevance

We call corpus the collection of all documents which we will note $\mathcal{C}$. The set $\mathcal{R}_q(t,f,u)$ of all documents relevant to a need q is defined by:

$$\mathcal{R}_q(t,f,u) = \{d \in \mathcal{C} \mid d \text{ is relevant to } q \text{ for a given time } t \text{ and a given facet } f \text{ dedicated to a specific user } u\}$$

The set $\overline{\mathcal{R}_q}(t,f,u)$ is the set of all irrelevant documents of the corpus:

$$\overline{\mathcal{R}_q}(t,f,u) = \{d \in \mathcal{C} \mid d \text{ is irrelevant to } q \text{ for a given time } t \text{ and a given facet } f \text{ dedicated to a specific user } u\}$$

Hypothesis (1) and (2) introduce the basic notion of relevance and irrelevance. We complete them by the following hypothesis.

Hypothesis 5 (Absence of user judgment) *We admit that there could exist documents neither relevant or irrelevant to a need q given a user u.*

We note $\mathcal{U}_q(t,f,u)$ the set of these documents. The hypothesis (1) (2) and (5) can be expressed by the partition of the corpus $\mathcal{C}$ in three sets:

$$\mathcal{C} = \mathcal{R}_q(t,f,u) \cup \overline{\mathcal{R}_q}(t,f,u) \cup \mathcal{U}_q(t,f,u)$$

Our notion of facet is linked to the notion of relevance in the following way: if a user judges a document both relevant and irrelevant at the same time, we understand this by the fact that this user has taken two different point of view for his judgment. Hence for a given facet we think the user's judgment must be consistent.

Hypothesis 6 (Static consistency) *At a time t, for a facet f and for a user u, there is no document judged both as relevant and irrelevant: $\mathcal{R}_q(t,f,u) \cap \overline{\mathcal{R}_q}(t,f,u) = \emptyset$. The user's judgment is said to be statically consistent at this time and for this facet.*

As user cannot obviously expresses both some judgment (of relevance or irrelevance) and no judgment at all, and with hypothesis static consistency (6), existence of user relevance (1), existence of user irrelevance (2), and absence of user judgment (5) we can establish the following.

$$\mathcal{R}_q(t,f,u) \cap \overline{\mathcal{R}_q}(t,f,u) \cap \mathcal{U}_q(t,f,u) = \emptyset$$

In a "normal" session, users can only discover new relevant documents in addition to those previously judged as relevant. So a change of relevance judgment during time is interpreted as an user error.

For a given document d and a given user u, and for two times $t_1 < t_2$, possible evolutions are:

1. If $d \in \mathcal{U}_q(t_1, f, u)$, and later $d \in \mathcal{R}_q(t_2, f, u)$ or $d \in \overline{\mathcal{R}_q}(t_2, f, u)$: this means that at t_1 the user does not know the relevance of d, but he has made a judgment at t_2. For example, he could discover the document, read it and then have an opinion about it.
2. If $d \in \mathcal{R}_q(t_1, f, u)$ and $d \in \overline{\mathcal{R}_q}(t_2, f, u)$, or $d \in \overline{\mathcal{R}_q}(t_1, f, u)$ and $d \in \mathcal{R}_q(t_2, f, u)$: in this case, the user simply change his mind about the document.
3. If $d \in \mathcal{R}_q(t_1, f, u)$ or $d \in \overline{\mathcal{R}_q}(t_1, f, u)$ and later $d \notin \mathcal{R}_q(t_2, f, u)$ and $d \notin \overline{\mathcal{R}_q}(t_2, f, u)$, this means that the user either has forgotten a judgment or that he is confused and does not know the right judgment about document d.

Aside situations where at time t_2 a document remains in the same class as it was at time t_1, these are all the *possible* evolutions based on this temporal approach. We propose now to describe some characteristics to qualify the evolution of relevance judgments in time.

During a retrieval session, when the user states some relevance judgment and do not change his mind times after, we say his judgment is *stable* over time. Moreover, we say that the user judgment is *consistent* when he does not state a relevance judgment at time t_1 and its opposite judgment later at time t_2 for the same document.

Definition 3 (Stability of relevance judgment) *For $t_1 < t_2$, if $\mathcal{R}_q(t_1, f, u) \subseteq \mathcal{R}_q(t_2, f, u)$ then we say that the user relevance judgment is stable.*

Definition 4 (Stability of irrelevance judgment) *For $t_1 < t_2$, if $\overline{\mathcal{R}_q}(t_1, f, u) \subseteq \overline{\mathcal{R}_q}(t_2, f, u)$ then we say that the user irrelevance judgment is stable.*

If the user judgment is not stable, then it is desirable that it will be at least consistent on time. This means that if a positive judgment is made at a certain time, then this judgment can only be revised to become unknown, but it cannot be transformed into a negative judgment.

Definition 5 (Dynamic consistency) *For $t_1 < t_2$, if $\mathcal{R}_q(t_1, f, u) \cap \overline{\mathcal{R}_q}(t_2, f, u) = \emptyset$, then we say that the user judgment is consistent over time.*

The definitions (6), (3) and (4) imply definition (5). The reverse implication is false.

7.2.4 Facet dependent property

Some facets can be related in the relevance relation. For example, the facet that deals with the represented shape of an image is related to the facet that deals with the symbolic view. We propose at this time, only one case of facet-dependant property.

Definition 6 (Facet dependence) *For two different facets f_1, f_2 at time t where*

$$\mathcal{R}_q(t, f_1, u) \cap \mathcal{R}_q(t, f_2, u) \neq \emptyset$$

then these two facets are declared relevant-dependant at this time. Otherwise they are declared relevant-independent facets.

7.2.5 Order of relevance

In most existing systems, retrieved documents are ranked by relevance. It is assumed that a user is able to distinguish documents that are more relevant than others. Relevance decisions may be based on very different reasons because the user may be interested in different aspects of the document.

In [Yao, 1995] Yao introduce several notions of *user preference* using the relation $\succ$ defined as:

$$\text{For } d, d' \in \mathcal{C},\ d \succ d' \Longleftrightarrow \text{the user prefers } d \text{ to } d'.$$

This relation is called a *strict preference relation* and is defined as a subset of the Cartesian product $\mathcal{C} \times \mathcal{C}$. This preference relation is *partial*: some documents d and d' may be incomparable because it "does not make sense to compare them from the user point of view", says Yao. He introduces the indifference relation $\sim$ defined as:

$$d \sim d' \Longleftrightarrow (\neg(d \succ d') \wedge \neg(d' \succ d)).$$

We can instantiate these two relations with our formalism in the way suggested in [Yao, 1995]. At a time t, considering a facet f and a user u:

$$d \succ_t d' \Longleftrightarrow (d \in \mathcal{R}_q(t, f, u) \wedge d' \in \overline{\mathcal{R}_q}(t, f, u))$$

$$d \sim_t d' \Longleftrightarrow (\neg(d \succ_t d')) \wedge\ (\neg(d' \succ_t d))$$

Which is:

$$d \sim_t d' \Longleftrightarrow (d \notin \mathcal{R}_q(t, f, u) \vee d' \notin \overline{\mathcal{R}_q}(t, f, u)) \wedge (d' \notin \mathcal{R}_q(t, f, u) \vee d \notin \overline{\mathcal{R}_q}(t, f, u))$$

This means that two documents are indifferent in the sense of this relation, if they are not judged one relevant and the other irrelevant.

This definition supposes the use of the standard two-valued relevance scale. For a m-valued scale of relevance we have the following choice:

1. We split the set of document $\mathcal{C}$ into m classes $\mathcal{R}_q(t,f,u)_i$ each of which expressing a degree of relevance and the set noted $\mathcal{U}_q(t,f,u)$ expressing the class of documents that are indifferent to the user:

$$\mathcal{C} = \bigcup_i \mathcal{R}_q(t,f,u)_i \cup \mathcal{U}_q(t,f,u)$$

2. We split the set of document $\mathcal{C}$ into $m-1$ sets : $\mathcal{R}_q(t,f,u)_i$, plus the set of irrelevant documents $\overline{\mathcal{R}_q}(t,f,u)$ and the set of un-judged documents $\mathcal{U}_q(t,f,u)$. In that case the preference relation has a meaning only in the set $\mathcal{R}_q(t,f,u)$.

$$\mathcal{C} = \bigcup_i \mathcal{R}_q(t,f,u)_i \cup \overline{\mathcal{R}_q}(t,f,u) \cup \mathcal{U}_q(t,f,u)$$

The first approach only expresses some degrees of relevance: irrelevance is not explicitly expressed. We prefer the second approach because it makes a clear distinction between what is relevant, and what is not relevant. Moreover, this allows to state that the user does not wants a given collection of documents to be retrieved later by the system.

As times becomes a parameter of the retrieval process, we include time into the preference relation. Facets of documents are associated to points of view a user can have about a document. It seems also important to include the facet notion into the definition of relevance.

Definition 7 (Set of relevant documents) *We note $\mathcal{R}_q(t,u)$ the set of relevant documents for at least one facet at time t and for a user u:*

$$\mathcal{R}_q(t,u) = \bigcup_f \mathcal{R}_q(t,f,u)$$

Hypothesis 7 (Relevance pre-order) *There exists a partial pre-order on relevance: at a time t, and for a user u, the set $\mathcal{R}_q(t,u)$ is partially pre-ordered.*

Users are more interested in document that are relevant to their queries than in documents considered as irrelevant. Consequently the relevance order is more important than the irrelevance order which can be useful for the system but not for the user. At least, the user could express he does not want the system to retrieve again a given document because

he knows it already well. This is one reason why it is important to take into account the irrelevance user judgment.

Under this hypothesis, it is assumed that a user is able to decide whether a document is more useful or more relevant than another document. The pre-order means that this relevance judgment is a reflexive and transitive relation on documents. This order cannot be total because there could exist documents that are not comparable from the user's point of view.

This order is not antisymmetrical because of the facet notion: there exist documents that are symmetrically ordered: $d_1 \succ d_2$ and $d_2 \succ d_1$. This means that the user has different points of view to rank documents according to two different facets. We can also use this notion to define the notion of facet using the preference order.

Hypothesis 8 (Facet existence) *Given a user u at time t, for two documents d_1, d_2, when the user both ranks $d_1 \succ d_2$ and $d_2 \succ d_1$ then we explain this situation by the fact that there exists two different facets f_1 and f_2, the user has used to differently rank the documents : $d_1 \succ_{f_1} d_2$ and $d_2 \succ_{f_2} d_1$.*

Using this hypothesis, for a given facet, the relevance pre-order is antisymmetric and thus is an order.

Hypothesis 9 (Facet relevance order) *There exists a partial order on relevance relation: at a time t, for a facet f, and a user u, the set $\mathcal{R}_q(t, f, u)$ is partially ordered.*

We could also consider the hypothesis of an order on the irrelevance relation but, as said before we think this order is less important than the one for relevance relation because the goal of an IRS is to emphasis the structure of the relevant documents and not the structure of the irrelevant documents.

Hypothesis 10 (Irrelevance pre-order) *There exists a partial pre-order on the irrelevance relation: at a time t, given a user u, the set $\overline{\mathcal{R}_q}(t, u)$ is partially pre-ordered.*

7.2.6 Time and facet-related definitions

When considering the previous notions of time variation of relevance and of partial order associated to the evaluation of relevance at any given time, one can address some new interesting notions that describe how the retrieval process evolves over time. An important one is the notion of convergence that expresses whether or not the retrieval process has reached a stable state (i.e. there are no changes about retrieved documents and there ranking from time t_1 to time t_2 despite all the intermediary interactions that occurred in this interval).

When taking into account the partial order, we can talk about the *convergence* of the search. This means that a retrieval process converge toward some documents if the ranking of relevance has some stability. As we use a partial order, we talk about strong or weak convergence when the relation order holds over time or not.

Definition 8 (Weak temporal convergence) *Given a partial order on $\mathcal{R}_q(t,f,u)$, for all $d_1, d_2 \in \mathcal{R}_q(t_1,f,u)$ with $d_1 \succ_{t_1} d_2$, if for all sets $\mathcal{R}_q(t_2,f,u)$ where $t_2 > t_1$, one have $d_1, d_2 \in \mathcal{R}_q(t_2,f,u)$ and d_1 and d_2 are comparable, and one still have $d_1 \succ_{t_2} d_2$, then we say that the sequence of sets $\mathcal{R}_q(t,f,u)$ weakly converge.*

This property expresses that if a given partial order exists at time t_1 among relevant documents, part of this order still exists at time t_2 and this order is not contradictory with the order at time t_1

Definition 9 (Strong temporal convergence) *Given a partial order on $\mathcal{R}_q(t,f,u)$, for $d_1, d_2 \in \mathcal{R}_q(t_1,f,u)$ with $d_1 \succ_{t_1} d_2$, if for all $\mathcal{R}_q(t_2,f,u)$ where $t_2 > t_1$, $d_1, d_2 \in \mathcal{R}_q(t_2,f,u)$ then d_1 and d_2 are comparable and $d_1 \succ_{t_2} d_2$ then we say that the sequence of sets $\mathcal{R}_q(t,f,u)$ strongly converge.*

This last definition states that if there exists a partial order among relevant documents at time t_1, then this order still holds at time t_2.

7.2.7 Consistency between abstract and actual relevance

If the abstract relevance exists, it is reasonable to think that the user is consistent with it. We can say that the user actual relevance is a way of approaching the abstract relevance over time.

Hypothesis 11 (Consistence of the actual relevance with abstract relevance) *We suppose that a user in an actual situation has a relevance judgment that is consistent with the abstract one.*

For all facets, this can be expressed by:

$$\forall t, \mathcal{R}_q(t,u) \subseteq Rel(\mathcal{C},q) \text{ and } \overline{\mathcal{R}_q}(t,u) \subseteq \overline{Rel}(\mathcal{C},q)$$

This hypothesis implies the following property.

Property 1 (Consistency of relevance) *For all t and t', for a given facet f and a given user u:*

$$\mathcal{R}_q(t,f,u) \cap \overline{\mathcal{R}_q}(t',f,u) = \emptyset.$$

In return, as long as this property is verified, the hypothesis (11) is not invalidated by the user's behavior.

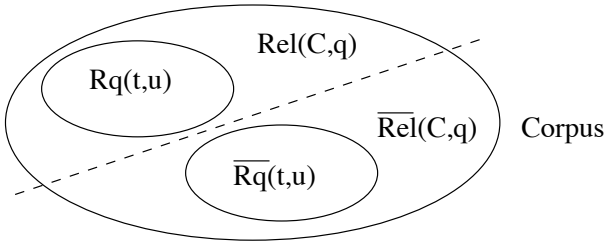

Figure 7.2: Actual user relevance

7.3 System relevance

Until now, we have ignored the retrieval system and concentrated on the notion of user relevance. When introducing the IRS in the retrieval process, we have to be able to describe in a formal way its behavior compared to user's needs.

Hence, objective of IR models is to be able to formalize the informational content of d, the subjective user's need q and how the system will help to match d and q in an appropriate way. The following principle is the ground of an IR modeling.

Hypothesis 12 (IR processing is formalizable) *There exist formalisms in which d and q can be partially expressed within the relevance relation. We call the formalization of d the index noted $IND(d)$ or simply D, and we call the formalization of q the interpretation of the query noted $INT(q)$ or simply Q. The retrieval mechanism of system s can compute the set of all documents $\mathcal{R}_Q(t,s)$ that matches query Q at time t. There also exists in the model, a matching relation noted $\cong$ between D and Q.*

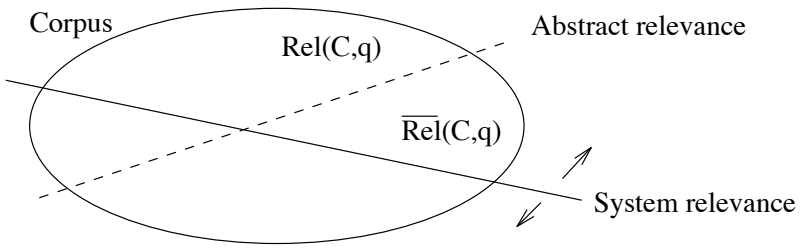

Figure 7.3: System relevance

We say that the formalism *partially* expresses the original objects because the information in D is a degradation (i.e. an approximation) of the original information in d and because the original user's need q is also approximated in Q. We think that considering

these degradations is one key point of an effective and complete IR model: along with the quality of the matching function, this explain why retrieved documents may not be always relevant to the user, and why relevant documents may not be retrieved at all. This introduce the well known notion of system relevance, as opposed as the one of user relevance introduced before.

Documents that are system relevant for a given query Q and a system s, are noted:

$$\mathcal{R}_Q(t,s) = \{d \in \mathcal{C} | D = IND(d) \wedge D \cong Q \wedge time(Q) = t\}.$$

This is the set of documents the system s judges d as relevant to the query Q at time t.

The system applies its matching function to all documents of the corpus. Each document of the corpus is then either relevant or irrelevant:

Axiom 1 (System relevance) *All IRS are defined in a way that :*

$$\mathcal{R}_Q(t,s) \cup \overline{\mathcal{R}_Q}(t,s) = \mathcal{C} \text{ and } \mathcal{R}_Q(t,s) \cap \overline{\mathcal{R}_Q}(t,s) = \emptyset$$

At any moment when the query is processed, system relevance induces a partition of the corpus and this partition can evolve during the retrieval process. The axiom (1) is the equivalent of the static consistency hypothesis (6) from the system point of view. Other characteristics of user relevance cannot be applied for system relevance because with this axiom we state that the system is *always* able to compute the relevance for *all* documents, *all* the time. As the user relevance judgment is taken as a reference, if there is a disagreement between the user and the system, the latter must change it's point of view. In this context, stability and consistency of the system relevance judgment is a non sens.

We have supposed the user able to rank the received documents according to their relevance judgment. This order is necessarily partial because a normal user cannot know all documents of the base in order to completely rank them, and moreover it could exist documents that are not comparable from the user's point of view. On the contrary, the system can always establish an absolute ranking of the documents, and this ranking is often total. In many system the result of the matching between the query and documents is resumed by real number that express the relevance ranking.

In figure (7.3) we show that the system relevance splits the corpus in a different way that the abstract relevance does. The goal of the system is to help to reach a state where the system relevance is as close as possible to the abstract relevance. To achieve this goal, the system can use user feedback and query reformulation.

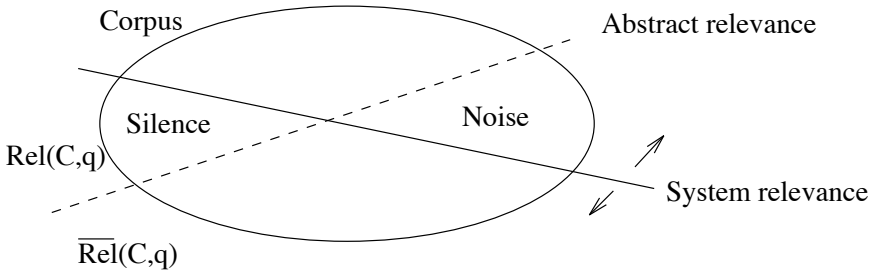

Figure 7.4: Abstract silence and noise

7.3.1 Silence and noise

With these notations, an ideal information retrieval system is the one that selects the same documents that the user would select:

$$\forall t,\ \mathcal{R}_Q(t,s) = Rel(\mathcal{C}, q)$$

This evaluation of this condition cannot be computed in practice because the abstract user relevance $Rel(\mathcal{C}, q)$ cannot be reached in an normal interaction. First, because the need q has to be expressed into Q in the query language of the IR system and second, because it would be infeasible and absurd to ask the user to check all the documents and tell the system which ones are relevant. In test collections $Rel(\mathcal{C}, q)$ is estimated for a given set of user's need q.

The **abstract silence** is the theoretical set of documents that are relevant but not retrieved by the system.

Definition 10 (Abstract silence) *At time t, the abstract silence is defined by*

$$Rel(\mathcal{C}, q) - \mathcal{R}_Q(t,s).$$

The overall abstract silence for a whole retrieval session, is defined by

$$Rel(\mathcal{C}, q) - \bigcup_t \mathcal{R}_Q(t,s).$$

The abstract noise are documents retrieved by the system but irrelevant to the user's need.

Definition 11 (Abstract noise) *At time t, the abstract noise is defined by*

$$\mathcal{R}_Q(t,s) - Rel(\mathcal{C}, q).$$

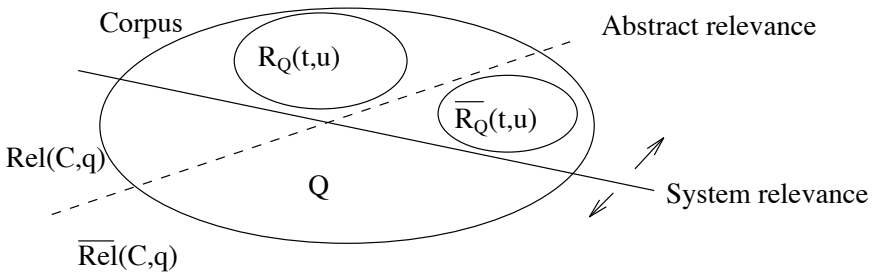

Figure 7.5: First user feed back

The overall noise for an entire retrieval session, is defined by

$$\bigcup_t (\mathcal{R}_Q(t,s) - Rel(\mathcal{C},q)).$$

7.3.2 User feedback

User feedback is an information the system has to take into account to adapt its matching computation. This adaptation can be either a modification of the query Q, some modifications of the document's index D, or the modification of the matching evaluation itself. Most often in practice, the query Q is the only information expanded using knowledge from a thesaurus and using documents selected by the user as relevant (positive feedback) or as irrelevant (negative feedback).

At the first feedback step at time $t = 1$, the user can only assess the relevance or irrelevance of documents retrieved by the system according to his original query Q submitted at time $t = 0$. At a time t user select documents on a list proposed by the system. Hence we have the following relations.

- The user selects some relevant documents : $\mathcal{R}_q(t,u) \subseteq \mathcal{R}_Q(t,s)$.
- The user selects some irrelevant documents : $\overline{\mathcal{R}_q}(t,u) \subseteq \mathcal{R}_Q(t,s)$.

In general, in a particular step of relevance feedback, at time t the system store the user's feedback of all previous user's answer at time $t' < t$. In the following we analyze all possible sets intersections.

1. $\mathcal{R}_Q(t,s) \cap \mathcal{R}_q(t',u)$: this is the set of relevant documents that are at time t retrieved by the system and previously accepted as relevant by the user.
2. $\overline{\mathcal{R}_Q}(t,s) \cap \overline{\mathcal{R}_q}(t',u)$: this is the set of documents that the user does not want to see anymore and that the system has not retrieved.

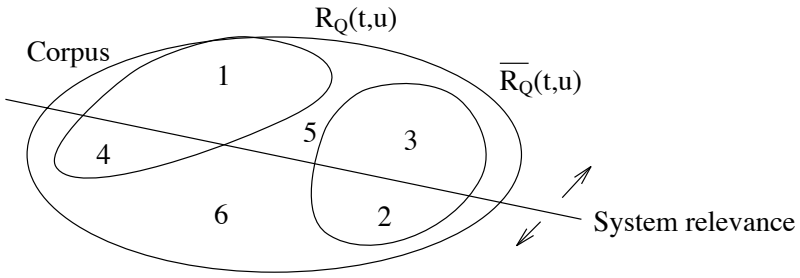

Figure 7.6: General situation of feed back

3. $\mathcal{R}_Q(t,s) \cap \overline{\mathcal{R}_q}(t',u)$: this is the set of documents that the system still retrieves whereas the user does not want then. This corresponds to a dynamic noise.

4. $\overline{\mathcal{R}_Q}(t,s) \cap \mathcal{R}_q(t',u)$: this is the set of documents that the user has noted relevant to the system and that the system is no more able to retrieve at time t. We call this set dynamic silence.

5. $\mathcal{R}_Q(t,s) - \mathcal{R}_q(t',u) - \overline{\mathcal{R}_q}(t',u)$: this is the set of documents retrieved for which the user does not have any opinion.

6. $\overline{\mathcal{R}_Q}(t,s) - \mathcal{R}_q(t',u) - \overline{\mathcal{R}_q}(t',u)$: this is the set of not retrieved document for which the user has any opinion.

Definition 12 (Feedback quality) *For $t > t'$, feedback dynamic quality is measured in terms of:*

- $\overline{\mathcal{R}_Q}(t,s) \cap \mathcal{R}_q(t',u)$ *that expresses dynamic silence;*
- $\mathcal{R}_Q(t,s) \cap \overline{\mathcal{R}_q}(t',u)$ *that expresses dynamic noise.*

The feedback quality is high when the size of these two sets are low.

7.3.3 Matching relation

In this part, we study possible matching relations, their properties and consequences on retrieval efficiency. At first, we can examine several possible definitions of the indexation relation IND. This relation associate real documents of the corpus $\mathcal{C}$ to index. We propose to examine two cases that we will use later. In the first case, only one index is associate to documents. We call it *mono indexing*. In the second we associate a set of index and call it *multi indexing*.

Definition 13 (Mono indexing) *For a given real document d there exists one and only one index D associated by the relation IND.*

Definition 14 (Multi indexing) *For a given real document d there exists one and only one set of index D associated by the relation IND.*

In both cases, we can build a system that ensure the retrieval of only one document given an index. If this property is not verified then the system is not able to retrieve separately two different documents that have the same index. This can be a problem if a user judges one as relevant and the other as irrelevant. We have two possible definition depending if we consider index or set of index.

Definition 15 (Discriminance of index) *Given an index D, it corresponds to at most only one real document d by the reverse indexing relation.*

Definition 16 (Discriminance of set of index) *Given a set of index $\{D_i\}$, it corresponds to at most only one real document d by the reverse indexing relation.*

On the query side, we can also consider individual query or sets of queries associated to a single user's need q. This situation typically arises when processing relevance feedback which leads to query reformulation that may be considered as new expressions of the same user's need q, provided that it has not changed. So expression Q evolves during the retrieval session but is always associated to an unique need q. As a result $INT(q)$ is the set of all different formalizations of the original user's need. This set is of course not entirely known from the system, but is incrementally obtained through the interaction.

Hypothesis 13 (Interpretation of user's need) *A user's need q is associated to a set of queries though the interpretation function: $INT(q) = \{Q_i\}$*

A good IR system aims to approximate user's relevance using a set of formulated and reformulated queries. This reformulation of Q can be done either by the system (automatic reformulation) in accordance to the feedback recorded from user's reactions, or by the user.

Hypothesis 14 (Relevance feedback) $\mathcal{R}_q(t,u) = \bigcup_{Q \in INT(q)} \mathcal{R}_Q(t,s)$

In order to formalize the notion of relevance we have to chose an adequate relation that models the relevance relation. We can propose for that a two-valued function that expresses whether the index D and the query Q match or not. For some systems (for example the vector space model) the matching function is multi-valued. For this modelization, we propose to define a two-valued matching function as a relation $\cong$ between D and Q.

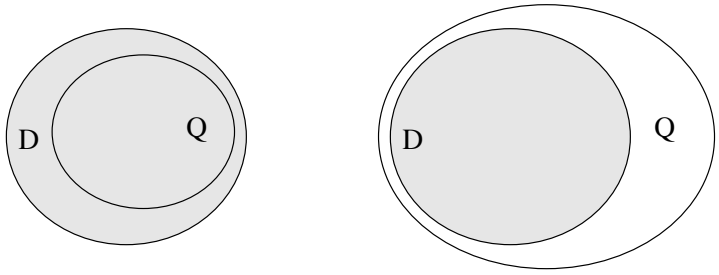

Figure 7.7: Matching criteria

Definition 17 (Matching relation) *We define a matching relation noted* $\mathcal{M}_q$ *by:*

$$\mathcal{M}_q =_{def} \{(D, Q) | Q \in INT(q) \vee D \cong Q\}.$$

A multi-valued matching function can be easily obtained by ordering the set of couples of $\mathcal{M}_q$. When we need a numerical total ordering we have to map all the couples from the relation $\cong$ to the interval of reals $[0, 1]$.

In [Nie and Chiaramella, 1990] Nie has proposed to split the matching function into two sub functions he called direct and reverse implication. As we do not have mentioned any logical notion at this step of our model, we split the matching relation into a direct matching $\overrightarrow{\cong}$ and a reverse matching $\overleftarrow{\cong}$.

Hypothesis 15 (Reverse and direct matching) *Modelization of* D *and* Q *are not necessary expressed in the same formalism. We suppose that the IR matching relation* $\cong$ *can be split into two relations:*

- *a direct matching relation from* D *to* Q *(*$D \overrightarrow{\cong} Q$*),*
- *a reverse matching relation from* Q *to* D *(*$Q \overleftarrow{\cong} D$*).*

The matching relation is computed by a combination of the direct and reverse matchings.

We can give an informal semantic definition of these relations:

- **Exhaustivity criteria**: $D \overrightarrow{\cong} Q$ means that D satisfies all the themes of Q.
- **Specificity criteria**: $Q \overleftarrow{\cong} D$ means that D only contents themes related to Q. In that sense, D is entirely dedicated to Q.

Definition 18 (Homogeneity of the model) *An IR model is* **homogeneous** *if the model of the real document and of the need, are both expressed in the same formalism. In that case, D and Q belong to the same set. In the reverse, we say that the model is* **heterogeneous**.

In some model, the reverse matching relation is exactly the same relation as the inverse direct matching relation. We call this property the symmetry of the matching relation.

Definition 19 (Symmetry of the matching relation) *An homogeneous model is symmetric when the reverse matching relation is identical to the inverse direct matching relation: $\not\cong = \not\cong^{-1}$.*

We recall the definition of an inverse relation:

$$\not\cong^{-1} =_{def} \{(Q, D) \mid D \not\cong Q\}$$
$$\text{or also}$$
$$Q \not\cong^{-1} D =_{def} D \not\cong Q$$

When the IR model is homogeneous, one can study general properties of the matching relation. This has been done in some studies like [Huibers and Denos, 1995; Huibers and Bruza, 1994]. For example, we can classify the matching relations ($\cong, \not\cong, \not\cong$) considering symmetry, reflexivity or transitivity.

7.3.4 Matching classification

In this part we propose to detail the matching function $\cong$ in term of comparing elements of D that we call *indexing terms*. These terms can be either atomic datum (such as keywords) that express *partially* document content or global interpretation of documents that expresses *globally* document content. We propose also some criteria to classify IR model. These two choices are dual. In practice, index terms are keywords or, in more recent and experimental systems they are more complex objects (a formula of a logic, a graph, etc). These models can be considered as generic in the sense that we do not need to know into details how to identify two descriptors, and how to decide when they are equal.

In the following list of models, we propose a classification based on set theory. In the following formulas, letters x, y are some index terms, n is an integer, and the function $Card(A)$ returns the cardinality of the set A.

1. **Equivalence matching:** The modelization here is simple: there exists an equivalent relation on indexing terms noted $=$. We use one simple indexing term for a query and for a document. When index terms are keywords, this model is uninteresting,

but when they are more complex objects, this is just like a data-base behavior which is an exact matching. For this model, the comparison is only an identity. This modelization is homogeneous.

$$Q =_{def} x,\ D =_{def} y$$
$$\cong =_{def} \{(D,Q) \mid Q = D\}.$$

2. **Membership matching**: The index document is a set of index terms. This model is heterogeneous.

$$Q =_{def} x,\ D =_{def} \{y_i\},$$
$$\models\!\cong =_{def} \{(D,Q) \mid Q \in D\}.$$

In theses models, there is often no difference between the reverse matching relation $\cong\!\models$ and $\models\!\cong^{-1}$, and the matching relations defined in terms of the direct matching relation : $\cong =_{def} \models\!\cong$.

3. **Inclusion matching**: Both document and query are sets of indexing terms. This model is necessarily homogeneous.

$$Q =_{def} \{x_i\},\ D =_{def} \{y_i\}$$
$$\models\!\cong =_{def} \{(D,Q) \mid Q \subseteq D\}$$
$$\cong\!\models =_{def} \{(Q,D) \mid D \subseteq Q\}$$

For the matching relation,we can choose for example among:

$$\cong =_{def} \{(D,Q) \mid D \models\!\cong Q \text{ or } Q \cong\!\models D\}$$
$$\cong =_{def} \{(D,Q) \mid D \models\!\cong Q \text{ and } Q \cong\!\models D\}$$

4. **Overlap matching**: In this type of model, the matching function measures the degree of overlapping between the document and the query.

$$Q =_{def} \{x_i\}, D =_{def} \{y_i\}$$

Different matching functions can be expressed using the intersection measure. For example, one can have a cut off value n to decide the matching.

$$\cong =_{def} \{(D,Q) \mid Card(Q \cap D) > n\}$$

All these models can rank the retrieved documents. For example, one can use the size of the overlap:

$$(D_1 \cong Q) \succ (D_2 \cong Q) \Leftrightarrow_{def} Card(Q \cap D_1) \geq Card(Q \cap D_2)$$

This classification makes sense because all models listed in [Blair, 1990] belongs to one of this four classes.

7.3.5 Integration of facet in system relevance

We have proposed a notion of facet to explain the preference of the user when evaluating the relevance of documents, and especially to explain some inconsistencies of his ranking choices. This notion of facet can be usefully introduced into the retrieval model. If the system can take into account some user's point of view, we can expect some improvement in the results. These improvements will be more important if the system facets are close to the users ones. We will be however limited to a fixed number of predefined facets according to the kind of media managed by the IRS.

Axiom 2 (System relevance and facets) *Indexes D and queries Q are split into a finite fixed number of facets f. Several matching functions $\cong_f$ are build and associated to the corresponding queries Q_f and document description D_f.*

Queries Q and indexes D are defined as the set all faceted queries and faceted indexes.

$$D = \{D_f\} \text{ and } Q = \{Q_f\}$$

We introduce now the set of documents selected by the system using facets.

$$\mathcal{R}_Q(t, f, s) = \{d \in \mathcal{C} | \text{ the system } s \text{ judges } d \text{ as relevant to the query } Q \text{ for a facet } f \text{ at time } t\}.$$

This can be expressed using the matching relation.

$$\mathcal{R}_Q(t, f, s) = \{d \in \mathcal{C} | D_f \in IND(d) \text{ and } D_f \cong_f Q_f\}$$

The set $IND(d)$ represent now all the index associated to one document. Each index of this set describe a particular facet of the document.

The set of retrieved document without using the facet notion can be defined in a first approach by a simple union on the facets.

$$\mathcal{R}_Q(t, s) = \bigcup_f \mathcal{R}_Q(t, f, s)$$

If all facets are independent, the set of all faceted indexes $\{D_f\}$ can then be structured by an **equivalence relation** defined by the membership of each D_f to a global index D.

Definition 20 (System facet dependence) *If two different facets f_1, f_2 verify the following condition at time t:*

$$\mathcal{R}_Q(t, f_1, s) \cap \mathcal{R}_Q(t, f_2, s) \neq \emptyset$$

then these two facets are declared relevance-dependent at this time. Otherwise they are declared relevance-independent facets at this time.

7.4 Application to a logic based model

In this section, we propose to employ the previously detailed general IR description to some logic-based models. At first, we start from the boolean model that we extend in order to include the notion of facet. Then, we propose a general framework using modal logic.

7.4.1 Modeling using a formalized logic

Until now, we have formalized IR relevance in a general framework without using any logic. Here we propose a strong assumption that will limit the formalization to only some logic. At the end of a successful retrieval process, the user has selected a relevant document and can always explain the reasons of his choice even if he's wrong (after reading carefully the document he may change his mind). So we make the following assumption, we believe that the notion of relevance between a document and a query deals with some logic. An example of logic was given previously with the boolean model.

Hypothesis 16 (Relevance is logic) *We make the assumption that for a document to be relevant to a query there exist a causal chain of deductions beginning at the document and ending at the query.*

The system relevance can then be proved by demonstrating that $D \rightarrow Q$. This implication is called the **information retrieval implication**. We examine here general principles one can assert concerning IR. We follow here ideas proposed by VanRijsbergen in [van Rijsbergen, 1986a] and developed by Nie in [Nie and Chiaramella, 1990].

The index D of a document d partially describes his informational content. There are many ways for a human being for reading, looking, hearing a document, so that it is obviously an even harder task for a computer to decide about "document content". In the same way, a query Q expresses only a part of q which is what the user wants: the user cannot express all his needs in details mainly because of the language barrier. An other obvious reason is that he usually cannot actually describe an information he is precisely looking for.

7.4.2 The boolean model

In the boolean model, D is a set of index terms $\{x_i\}$. The query Q is any boolean expression of index terms. Thus Q is a formula in the proposition calculus. The matching function $\cong$ is defined by:

- if $Q = x_i$ then $D \cong Q$ iff $x_i \in D$
- if $Q = Q' \vee Q''$ then $D \cong Q$ iff $D \cong Q'$ or $D \cong Q''$

- if $Q = Q' \wedge Q''$ then $D \cong Q$ iff $D \cong Q'$ and $D \cong Q''$
- if $Q = \neg Q'$ then $D \cong Q$ iff not $D \cong Q'$

If we change the phrase "$x_i \in D$" into "x_i is true in D", we obtain the classical definition of a logical interpretation of the formula Q in the proposition calculus. In an IR system, the interpretation of a term cannot be the usual one which is "the proposition denotes a fact that is true in this interpretation" because an index has nothing to do with truth values of the document content.

The only way that seems reasonable for interpreting atomic terms with a truth value is to consider the truth value associated to a term as the fact "the terms is a good descriptor of the document's content".

Definition 21 (Interpretation of index term truth) *For a given document d, we consider that this index is expressed by logical atomic indexing terms x_i. The interpretation of x_i is true if x_i is a good representation of the document d; x_i is false otherwise.*

Using this definition, we can say that a document index D is a logical interpretation of the indexing terms. In this way, the matching function is the logical consequence connector $\models$.

$$D \cong Q =_{def} D \models Q$$

This model is heterogeneous because D is a logical interpretation of atomic terms, and Q is a formula belonging to the logic language. In $D \models Q$, Q can also be understood as sets of interpretations. Thus $D \models Q$ is understood as "the interpretation D is in the set of interpretations Q". Thus, this model fits in the **membership matching**.

In propositional calculus, the semantic deduction theorem holds:

$$A \models B \text{ iff } \models A \supset B$$

Thus, in the boolean model, we can either consider D as an interpretation of index terms or as a logical formula. The matching function can also either be expressed by the semantic deduction, or the material implication. In this last case, we have to compute the validity of the formula $D \supset Q$. We have shown here that there are two possible equivalent modeling, the modeling based on validity and the modeling based on satisfiability in a logic with the semantical approach.

The interpretation of a document D is closely related to the notion of facet we have proposed in section (7.3.5) because facets are defined as different points of view about documents: one interpretation is then a particular point of view about a document.

$$\mathcal{R}_Q(t, f, s) = \{d \in \mathcal{C} \mid D_f \in IND_f(d) \text{ and } D_f \models Q_f\}$$

7.4.3 Extended boolean model

In the notation $D \models Q$, D can be either a set of interpretations or a formula. When D is a formula, $\models$ means that all interpretations of D validate the formula Q. Now $D \models Q$ means that D answers to Q over all facets which is a strong restriction to the matching process. Moreover, it is better to accept successful matching for at least one facet. We cannot express this refinement in classical propositional logic.

In modal logic, we can use the operator $\Diamond$ and $A \models \Diamond B$ that precisely means that B is true in at least one interpretation accessible from A. In our context, interpretations are associated to facets, and the accessibility relation links several facets associated to one global index D. As we have chosen an equivalence structure for the set $\{D_f\}$ the accessibility relation is an equivalence relation. The modal system S5 is based on a equivalence accessibility relation: it seems then to be the appropriate system to model a faceted boolean retrieval system.

For every document that belongs to $\mathcal{R}_Q(t,s)$ there exists a facet f so that this document belongs also to $\mathcal{R}_Q(t,f,s)$. This is true because $\mathcal{R}_Q(t,f,s)$ is defined as the union of $\mathcal{R}_Q(t,f,s)$. Thus, there exists an interpretation D_f so that $D_f \models Q$. We can then propose the following definition:

$$\mathcal{R}_Q(t,s) = \{d \in \mathcal{C} \mid D = IND(d) \text{ and } D \models \Diamond Q\}$$

7.4.4 A modal retrieval model

In this part we focus on the use of a modal logic to express the relation $\cong$ or relations $\overset{\cdot}{\cong}$ and $\overset{\cdot}{\ncong}$ as a logical implication. We introduce the use of modality that better expresses the IR paradigm than classical logic does.

Lets recall some basic definitions [1]. A valuation V of a formula, is a mapping from the propositions *Prop*, to a set of logical values **2**. We usually use a set of two values and so we use a two valued logic:

$$V : Prop \rightarrow \mathbf{2}$$

We say that a given valuation models a formula, or is a *model* of a formula, if this formula is true using this valuation.

We are using now several models that are called *worlds*. The modality appears when these worlds are linked with a binary relation. We refer to this set of worlds W and to the transitions δ as a *transition structure*, or simply as *structure*. When considering modal formulas, the valuation becomes:

[1] For a definition of modal logic see [Chellas, 1980], [Hughes and Cresswell, 1972] or [Popkorn, 1994]. We have used in this paper notations proposed in [Popkorn, 1994].

$$V : Prop \rightarrow \mathbf{PW}$$

$$\delta \subseteq W \times W$$

The valuation of a proposition is an element of the power set **PW** of W (a subset of W). It defines the subset of all worlds of W where a proposition is true. Relations between worlds express a modality that means that a world is accessible, or possible from an other world. The new operator $\Box$ called necessity, means that a formula $\Box\Phi$ is valid in a world w, if all the accessible worlds from w by δ make Φ valid:

$$w \models \Box\Phi \Leftrightarrow (\forall x)[w\delta x \text{ and } x \models \Phi]$$

If we combine the sets of **PW** by union, intersection and complementation

$$X \cup Y, X \cap Y, \neg X = W - X$$

we obtain a Boolean algebra. With the operator $\Box X$ defined by:

$$a \in \Box X \Leftrightarrow (\forall x)[a\delta x \Rightarrow x \in X]$$

the obtained structure is called a modal algebra. The $\Diamond$ operator called possibility, is deduced from $\Box$: $\Diamond\Phi = \neg\Box\neg\Phi$.

In the following section, we describe a use of a modal logic for modeling IR which is in fact an extension of the Boolean model of IR.

7.4.5 Modal indexing

We propose to use modal logic for IR in this way : documents d are associated to a subset of W. Each element of W is a possible interpretation of a document, and thus a possible facet.

$$D = IND(d)$$
$$IND(d) \subset W$$

User's needs q are associated to queries Q expressed as formulas built from proposition $Prop$ and logical connectors ($\wedge, \vee, \neg$, and $\Diamond$). The modal valuation V of formula associates propositions to a set of worlds, and using the reverse IND relation, a proposition is associated to a set of document facets.

The transition δ between worlds expresses the relationship between facets. We propose to use this relation to represent the way facets are associated to documents. Two document facets D_f and $D_{f'}$ are linked by δ if they are facets of the same document:

$$\text{For } D_f, D_{f'} \in W : \exists d \mid D_f, D_{f'} \in IND(d) \Leftrightarrow D_f \ \delta \ D_{f'}$$

We will examine later, the way we can use modality for the retrieval part.

7.4.6 Modal retrieval

Using modal logic, we state that the IR matching $D \cong Q$, is represented by the satisfaction of the formula Q in one world associated to d for a given facet f. In a modal logic, the satisfaction relation $\models$ is defined for a set of worlds W, for a given valuation V of the atomic propositions on these worlds, and from a given world w.

$$(W, V, w) \models Q$$

As we propose to fix the set of worlds and the valuation which represents the result of the indexation process, we will simplify the notation into:

- $w \models Q$ when we consider a world not associated to a facet index;
- $D_f \models Q$ when we consider a world related to a given facet.

The set of retrieved documents for a given facet is defined as the set of documents for which there exists a document interpretation D_f (i.e. interpreted via a facet f) so that the query Q is valid:

$$\mathcal{R}_Q(t, f, s) = \{d \in \mathcal{C} \mid D = IND(d),\ D_f \in D \text{ and } D_f \models Q\}$$

Using a modal logic, there are three ways of defining a logical consequence: from a given world, from any world of a given structure, and from any structure:

1. satisfaction from a word of a structure: $w \models Q$
2. satisfaction from any worlds of a structure: $(\forall w) w \models Q$
3. satisfaction from any structures: $\models Q$

All these three logical consequences have a meaning in an IR context :

1. If we find D_f so that $D_f \models Q$, then D is a good answer to Q for the facet f.
2. If we find D_f so that $D_f \models \Diamond Q$, then D is a good answer to Q for at least one other facet of D.

3. If we find D_f so that $D_f \models \Box Q$, then D is a good answer to Q for all other facets of D.

4. If we have $(\forall w)w \models Q$, then every D is a possible answer to Q. Thus this query is more general for the given corpus of documents and for the set of known facets. On the other side, if $(\forall w)w \models \neg Q$ then we say that Q is too specific for the given corpus and for the known facets; there exists no document nor document facet that is an answer to Q.

5. As we do not constrain the formulation of the formula Q, we could obtain the situation where Q is true (or false) whatever the choice of a set of worlds and the choice of a structure of these worlds. We note this tautology $\models Q$ (or $\models \neg Q$). In this case, we reject the query because it is obviously a useless query for the user.

7.4.7 Using modality in a query language

In the model we propose here, a query is associated to a formula and a document is associated to a set of worlds. We examine now the use of modality in this formula and consequently, we propose a modal query language for IR.

Let's recall at first the meaning of the classical logic connector in an IR context. A correct query is associated to a satisfiable formula. We avoid tautology or absurdity: formulas so that $\models Q$ (or $\models \neg Q$) are not accepted as queries because all documents (respect. no document) would be relevant to this query.

We consider now index terms t that occur in the query Q. We say that a term t indexes a facet f of a document d if we have $D_f \in IND(d)$ and $D_f \models t$.

- The semantics of the "and" connector is: $D_f \models t_1 \wedge t_2$, which means that $D_f \models t_1$ and $D_f \models t_2$. This means the trivial fact that the "and" is used to retrieve documents that are indexed by t_1 and t_2 considering a facet f.

- The semantics of the "or" connector is: $D_f \models t_1 \vee t_2$, which means that $D_f \models t_1$ or $D_f \models t_2$. In a query, this expresses a kind of *uncertainty* about these two terms: the user is not sure which of these two terms are important to him because he accepts as relevant documents that are indexed by one term or both.

- The negation operator is used to reject documents indexed by a term, but not to retrieve a *negated fact*. For example, an IRS can retrieve documents about planes and not about wings, but we cannot express a query searching for document about planes without wings. In this model, we have either $D_f \models \neg t$ or $D_f \models t$.

- The modal operator is used to refer other facets: $D_f \models \Diamond t$ means that there exists another facet for D where D is indexed by t. The $\Box$ being equivalent to $\neg\Diamond\neg$ by

definition, $D_f \models \Box t$ means that the term t indexes all the other facets associated to the document.

We can then combine and use the modal operators in a query. For example, we can ask the query:

$$(t_1 \wedge \Box t_2) \vee \Diamond t_3$$

Here we are searching for a document indexed by t_1 on one facet and indexed by t_2 on all other facets, or indexed by t_3 on one other facet.

This small example only shows that modality is a possible way to formalize the notion of facet relevance. This model is a first step that introduces the notion of facet into an IRS model. More development are necessary to precise and generalize this use of logic and specially the use of modal logic. This work will be carried out even if this task is meant to be finished.

In the next section, we give a short example about the modeling of dynamic user interaction. This interaction must not be limited to a relevance feedback but extended to all useful feedbacks a system could obtain from the user in order to match his information needs. We are very interested by user interaction modeling because we think it can lead to a major break through for IR systems. This approach will also be carried on in the FERMI project.

7.5 Modeling the dynamic process of IR

A most important aspect an IR model should include is the dynamic behavior between the system and the user that occur within a session. The purpose of this interaction is to to find documents that are closer to the original user's needs. For example, the system can directly ask the user to solve ambiguities and find correct interpretation of the user's need. After the query formulation, the system can also adapt its relevance judgments by asking the user to select some retrieved documents.

The standard logic can easily describe a static process of logic deduction, but fails to express unknown information and the dynamic aspects of Information Retrieval. Specially, we want to formalize the user behavior when he is interacting with the retrieval process.

7.5.1 An example

To illustrate our goal, we present an example of interesting modal calculus. We propose to model the behavior of an IRS that have to choose the relevant documents from a given query. We model this fact in a high level, and for this example, we do not detail the content of the query nor the index of documents. We just propose to consider the fact

that a document D_A is relevant to the query or that the document D_B is relevant. The question is very simple.

We wants the system to discover the pertinence of D_A using the answers of the user. We propose a scenario in which the system must deduce facts from the user's interaction.

In this example, we assume that the notion of relevance is absolute and that the user or the system can only know or ignore this fact. We also assume that the user's judgment is correct and absolute (we know that this assumption is a bit strong and does not take into account the fact the the user can revise his own judgment).

The system starts by showing document's titles to the user. The system wants to know more about these documents and specially the user relevance. It asks the user about the first document D_A. The user only answers that he knows about the relevance of A because he has read it entirely. Then the system wants to know more about D_B. The user answers that he does not know the pertinence of D_B because he has not read it.

Now the user wants to leave the system and it asks him before leaving if he is satisfied. The user say yes, because he has found a relevant document. With this information, the system knows that D_A is a relevant document for the user, but it cannot say anything about D_B.

For us, this answer is obvious, but how can the system deduce this fact ? It only knows that the user knows the relevance or irrelevance of D_A, and he does not know the relevance of D_B, and finally that the user thinks that D_A or D_B is relevant. If D_A were not relevant it could not be sure that at least one of the two are relevant, because he does not know about the relevance of D_B. So the system knows now that D_A is relevant. We can make the same reasoning for any number of documents.

If we wants to model this reasoning we have to model the fact that D_A is relevant or not, D_B is relevant or not, and the fact the the user or the system *knows or not knows* some facts. This point is very important because it makes the difference between classical logic deduction and modal logic deduction as we will show it in the following.

7.5.2 Formalization

Let's use A to say that D_A is relevant and $\neg A$ to say that D_A is not relevant (idem for B). Lets use the symbol K_s as an unary operator expressing what the system knows and K_u to express what the user knows. The fact that the system knows that the user knows the relevance of D_A is expressed by $K_s(K_u A \vee K_u \neg A)$. Then the fact that the system knows that the user does not know the pertinence of D_B is expressed by $K_s(\neg K_u B)$. Finally, the fact that the system knows that the user has found at least one relevant document, is expressed by $K_s K_u(A \vee B)$.

For the formal deduction, we uses the axioms and rules (modus ponens) of the predicate logic. We must add rules and axioms for the new operators K_x.

First, we assume that if something is false, we cannot know it:

$$\neg\Phi \supset \neg K_x\Phi$$

As it is a thesis in PC that :

$$(\neg X \supset \neg Y) \supset (Y \supset X)$$

We propose the axiom:

$$K_x\Phi \supset \Phi \tag{7.1}$$

We must also add the fact that if someone knows that if he has X then he can deduce Y, when he knows X then he also knows Y. This is in fact the distributivity of the K_x operator:

$$K_x(X \supset Y) \supset (K_xX \supset K_xY)$$

Finally, one must propose a rule that deals with K_x expressing that if something is true, then we know that it is true:

$$\frac{\Phi}{K_x\Phi} \tag{7.2}$$

With this system, we must prove that K_sA :

7.5.3 Proof

The system knows the fact that the user has found at least one relevant document :

$$K_sK_u(A \vee B) \tag{7.3}$$

This can be rewritten in :

$$K_sK_u(\neg A \supset B)$$

Using the distributivity axiom we have :

$$K_u(\neg A \supset B) \supset (K_u\neg A \supset K_uB) \tag{7.4}$$

We can add in any formula $X \supset Y$ the operator K_x and obtain $K_xX \supset K_xY$. If we have proved $X \supset Y$ by using rule (7.2) we obtain $K_x(X \supset Y)$; by using distributivity we have : $K_x(X \supset Y) \supset (K_xX \supset K_xY)$ and finally by modus ponens we prove that $K_xX \supset K_xY$. So we resume by the rules :

$$\frac{K_x(\Phi \supset \phi)}{K_x\Phi \supset K_x\phi} \tag{7.5}$$

$$\frac{\Phi \supset \phi}{K_x\Phi \supset K_x\phi} \tag{7.6}$$

With this new rule we can deduce from (7.4) :

$$K_s(K_u(\neg A \supset B)) \supset K_s(K_u \neg A \supset K_u B)$$

and by modus ponens with (7.3) we obtain :

$$K_s(K_u \neg A \supset K_u B) \tag{7.7}$$

The formula $(X \supset Y) \supset (\neg Y \supset \neg X)$ is an axiom of the proposition calculus. So is by rule (7.6) the formula $K_s(X \supset Y) \supset K_s(\neg Y \supset \neg X)$. By replacing X and Y by $K_u \neg A$ and $K_u B$, and using modus ponens with (7.7) we have :

$$K_s(\neg K_u B \supset \neg K_u \neg A) \tag{7.8}$$

Now let's use again distributivity (7.4) on (7.8) to build $K_s(\neg K_u B \supset \neg K_u \neg A) \supset (K_s \neg K_u B \supset K_s \neg K_u \neg A)$ and with modus ponens with (7.8) we obtain :

$$(K_s \neg K_u B \supset K_s \neg K_u \neg A) \tag{7.9}$$

The fact that the system knows that the user do not knows the pertinence of D_B is expressed by $K_s(\neg K_u B)$. So with the modus ponens on (7.9) we obtain :

$$K_s \neg K_u \neg A \tag{7.10}$$

The fact that the system knows that the user knows the relevance of D_A is express by $K_s(K_u A \vee K_u \neg A)$ that we can rewrite in $K_s(\neg K_u \neg A \supset K_u A)$. With the rule (7.5) we obtain $K_s \neg K_u \neg A \supset K_s K_u A$ and with modus ponens with (7.10) we obtain :

$$K_s K_u A \tag{7.11}$$

Finally with the axiom (7.1) we build $K_u A \supset A$ and deduce with (7.6) $K_s K_u A \supset K_s A$, by using at least the modus ponens with (7.11) we obtain what we wanted to prove, that is :

$$K_s A$$

The logic we have used is a propositional modal logic. More precisely, we have used the Kripke system with the characteristic formula :

$$\Box(P \supset Q) \supset (\Box P \supset \Box Q)$$

The system used is multi-modal using K_s and K_u unary operators. The modality used here enable to talk about knowledge of facts. As we have shown, it can be useful in IR system as a complete IR system must deal with user response.

We could build such a system using the (strong) assumption that the user knows the relevance of documents when he asks for reading more than the title. If the system only ask him at the end if he is satisfied or not, one could compute some knowledge as we have done in this example.

7.6 Conclusion

In this paper we have proposed a general framework to describe user and system relevance. We have introduced the notion of time related to user relevance as opposed to abstract relevance. We have also introduced the notion of user's point of view as expressed by the notion of facet. We have shown that this notion is closely related to the user ranking of preferred documents. We have also proposed an attempt to instantiate this approach into a logic chosen among the modal proposition logic. The obtained model does not refer to how the index terms are made, and these terms could not be only keywords, but any element of a more complex structure (ex: graphs).

This work is a first step in the understanding of the notion of relevance used in today Information Retrieval Systems. The approach we have followed is the definition of precise requirements about what seems important to model an IR system: user relevance versus system relevance, user interaction and relevance feedback.

Our goal was to make clear the underlying assumption IRS are based on. By making them explicit, we hope to reach a better understanding of IRS modeling, implementing and at least more effective performance measures. This was one of the possible directions we had mentioned in the previous FERMI report for task 1.3. We can now say that this goal is almost achieved.

The other possible direction was an in deep studies of the use of modal logic for modeling IR relevance. After having studied a lot of possible modal logic, we have still retain in this report a very small example of the use of these logics. In fact, at the beginning of the task of this FERMI project, we were convinced that the IR relevance notion would necessary go thought a modal fuzzy logic. After this exploration, we are not so categorical, and we better convinced that if we wants to bring a major improvement on the IRS behavior in the relevance side, we have to detail and to deeply express what is the perception of the notion IR relevance we use in our systems.

A new aspect of the relevance notion is emerging during this study: the modeling of time related user feedback. We think that this notion is tightly linked to the relevance notion. In this paper, we have just mention an example of modelization using logic. In fact, we have open an unexpected viewpoint and it will need the same formal approach we have taken to modelize user and system relevance.

In conclusion, we will still work on these directions on the FERMI project even if this task 1.3 is theoretically closed. We have given only an example of user interaction modelization, but time modeling is an important element for the dynamic of interaction and some more studies will be pursued in order to correctly specify these process.

Part III

Appendices

Appendix A

Existing Extensions to TLs

In this chapter we briefly describe the extensions, from an expressive power point of view, developed in the context of TLs. The list is not exhaustive but includes all features that may be of some interest to IR.

Probabilistic extension: Probabilistic versions of TLs [Hollunder, 1994; Sebastiani, 1994; Yen, 1991] could 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 domain and predicates on these domains [Baader and Hanschke, 1991]. Therefore, incorporating kinds of data types as "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 [Donini *et al.*, 1991b; Patel-Schneider *et al.*, 1991];

Closed World Assumption, Closed Domain Reasoning: Closed world reasoning and closed domain reasoning seem to be suitable for IR purposes, as they are close to usual databases reasoning [Donini *et al.*, 1992c; Reiter, 1990b; Reiter, 1978; Reiter, 1984];

Temporal extension: Integrating time into TLs using temporal logics and interval calculus, yielding a temporal TL which combines structural with temporal abstraction [Artale and Franconi, 1994; Artale and Franconi, 1995; Schmiedel, 1990];

N-ary terms extension: Usually, TLs allows the representation of at most two place relations. N-ary terms allows the representation of relations whose arity exceeds two [Schmolze, 1989];

OODBMS extension: Extension about the integration of TLs and Object Oriented Database Systems seems to be very useful for both [Beneventano *et al.*, 1993];

Relational database operators extension: The operators of relational databases are integrated into TLs and are shown to be very useful [De Giacomo and Lenzerini, 1995];

Modal extension: Modal operators are integrated into TLs, yielding a modal TL which handles notions as belief, intentions and time, which are essential for the representation of multi-agent environments [Baader and Laux, 1995];

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 TLs [Straccia, 1993].

Appendix B

Proofs

B.1 Proofs of Section 4.4

Lemma 2 *Let C, D two concepts and a an individual. Then $C \sqsubseteq D$ if and only if $\{C(a)\} \not\approx D(a)$.*

Proof:

$\Rightarrow$) Assume $C \sqsubseteq D$. Suppose that $\{C(a)\} \not\approx D(a)$. Therefore there is an interpretation $\mathcal{I}$ such that $t \in C^{\mathcal{I}}(a^{\mathcal{I}})$ and $t \notin D^{\mathcal{I}}(a^{\mathcal{I}})$. Therefore, $a^{\mathcal{I}} \in C^{\mathcal{I}}_{+}$, whereas $a^{\mathcal{I}} \notin D^{\mathcal{I}}_{+}$ and, thus, $C^{\mathcal{I}}_{+} \not\subseteq D^{\mathcal{I}}_{+}$, contrary to the assumption $C \sqsubseteq D$.

$\Leftarrow$) Assume $\{C(a)\} \not\approx D(a)$. Suppose that $C \not\sqsubseteq D$. It follows that there is an interpretation $\mathcal{I}'$ and $d \in \Delta^{\mathcal{I}'}$ such that $t \in C^{\mathcal{I}'}(d)$ and $t \notin D^{\mathcal{I}'}(d)$. Let $\mathcal{I}$ be an interpretation as $\mathcal{I}'$ except that $a^{\mathcal{I}} = d$. It follows that $t \in C^{\mathcal{I}}(a^{\mathcal{I}})$ and $t \notin D^{\mathcal{I}}(a^{\mathcal{I}})$, contrary to the assumption $\{C(a)\} \not\approx D(a)$. $\Box$

Lemma 3 *Each concept can be transformed into an equivalent NNF concept in polynomial time.* ∎

Proof: Let C be a concept. It is sufficient to apply repeatedly the following equivalences (in the following D, D' and R are concepts and a role, respectively):

1. $\neg\neg D \equiv D$;
2. $\neg(D \sqcap D') \equiv \neg D \sqcup \neg D'$;
3. $\neg(D \sqcup D') \equiv \neg D \sqcap \neg D'$;
4. $\neg(\forall R.D) \equiv \exists R.\neg D$;

5. $\neg(\exists R.D) \equiv \forall R.\neg D$. $\square$

Lemma 4 *No axiom is falsifiable. Equivalently, every axiom is valid.*

Proof: Let $\mathcal{I}$ be an interpretation and $\alpha, \Gamma \rightarrow \alpha, \Delta$ an axiom. If $\mathcal{I}$ satisfies $\{\alpha, \Gamma\}$ then $\mathcal{I}$ satisfies α, and thus $\mathcal{I}$ satisfies $\alpha, \Gamma \rightarrow \alpha, \Delta$. Since this is true for all $\mathcal{I}$, it follows that $\alpha, \Gamma \rightarrow \alpha, \Delta$ is valid. $\square$

Lemma 5 *For each rule in Definition 11, the conclusion of the rule is falsifiable iff at least one of the premises of the rule is falsifiable. Equivalently, the conclusion of the rule is valid iff all premises of the rule are valid.*

Proof: The proof consists in a case analysis on the rules of Definition 11. Remember that an interpretation $\mathcal{I}$ falsifies a sequent $\Gamma \rightarrow \Delta$ iff $\mathcal{I}$ satisfies all assertions in Γ and $\mathcal{I}$ does not satisfy any of the assertions in Δ.

case rule $(\sqcap \rightarrow)$:

$\Rightarrow$) Suppose that $\mathcal{I}$ falsifies the conclusion $(C \sqcap D)(t), \Gamma \rightarrow \Delta$. In particular, $\mathcal{I}$ satisfies $C(t)$, $D(t)$ and Γ. It follows that $\mathcal{I}$ falsifies $C(t), D(t), \Gamma \rightarrow \Delta$.

$\Leftarrow$) Suppose that $\mathcal{I}$ falsifies the premise $C(t), D(t), \Gamma \rightarrow \Delta$. It follows easily that $\mathcal{I}$ falsifies the conclusion $(C \sqcap D)(t), \Gamma \rightarrow \Delta$.

case rule $(\rightarrow \sqcap)$:

$\Rightarrow$) Suppose that $\mathcal{I}$ falsifies the conclusion $\Gamma \rightarrow \Delta, (C \sqcap D)(t)$. Therefore, $\mathcal{I}$ neither satisfies any of the assertions Δ nor satisfies $C(t)$ nor $D(t)$. In both cases, it follows that $\mathcal{I}$ falsifies at least one of the premises $\Gamma \rightarrow \Delta, C(t)$ and $\Gamma \rightarrow \Delta, D(t)$ of the rule.

$\Leftarrow$) Suppose that $\mathcal{I}$ falsifies $\Gamma \rightarrow \Delta, C(t)$ or $\Gamma \rightarrow \Delta, D(t)$. It is easy to see that in both cases $\mathcal{I}$ falsifies the conclusion $\Gamma \rightarrow \Delta, (C \sqcap D)(t)$.

case rule $(\sqcup \rightarrow)$: Can be proven as $(\rightarrow \sqcap)$.

case rule $(\rightarrow \sqcup)$: Can be proven as $(\sqcap \rightarrow)$.

case rule $(\rightarrow \forall)$:

$\Rightarrow$) Suppose that $\mathcal{I}$ falsifies the conclusion $\Gamma \rightarrow \Delta, (\forall R.C)(t)$. It follows that $\exists e \in \Delta^{\mathcal{I}}$ such that $t \in R^{\mathcal{I}}(t^{\mathcal{I}}, e)$ and $t \notin C^{\mathcal{I}}(e)$. Let $\mathcal{I}'$ be an interpretation as $\mathcal{I}$ such that $x^{\mathcal{I}'} = e$. It follows that $\mathcal{I}'$ satisfies Γ (note that x is a new variable). Since $t \in R^{\mathcal{I}}(t^{\mathcal{I}}, e)$ and $t \notin C^{\mathcal{I}}(e)$, $\mathcal{I}'$ falsifies $\Gamma, R(t, x) \rightarrow \Delta, C(x)$.

$\Leftarrow$) Suppose that $\mathcal{I}$ falsifies $\Gamma, R(t,x) \rightarrow \Delta, C(x)$. It then follows that $\mathcal{I}$ satisfies $R(t,x)$ and $\mathcal{I}$ neither satisfies any of the assertions in Δ nor $\mathcal{I}$ satisfies $C(x)$. Therefore, $\mathcal{I}$ falsifies the conclusion $\Gamma \rightarrow \Delta, (\forall R.C)(t)$.

case rule $(\exists \rightarrow)$**:**

$\Rightarrow$) Suppose that $\mathcal{I}$ falsifies the conclusion $(\exists R.C)(t), \Gamma \rightarrow \Delta$. It follows that $\exists e \in \Delta^{\mathcal{I}}$ such that $t \in R^{\mathcal{I}}(t^{\mathcal{I}}, e)$ and $t \in C^{\mathcal{I}}(e)$. Let $\mathcal{I}'$ be an interpretation as $\mathcal{I}$ such that $x^{\mathcal{I}'} = e$. It follows that $\mathcal{I}'$ falsifies $(\exists R.C)(t), \Gamma \rightarrow \Delta$ (note that x is a new variable) and satisfies both $R(t,x)$ and $C(x)$. Therefore, $\mathcal{I}'$ falsifies the premise $R(t,x), C(x), \Gamma \rightarrow \Delta$ of the rule.

$\Leftarrow$) Suppose that $\mathcal{I}$ falsifies the premise $R(t,x), C(x), \Gamma \rightarrow \Delta$. It follows that $\mathcal{I}$ satisfies $(\exists R.C)(t)$ and thus, $\mathcal{I}$ falsifies the conclusion $(\exists R.C)(t), \Gamma \rightarrow \Delta$ of the rule.

case rule $(\rightarrow \exists)$**:**

$\Rightarrow$) Suppose that $\mathcal{I}$ falsifies the conclusion $\Gamma \rightarrow \Delta, (\exists R.C)(t)$. It follows that $\mathcal{I}$ satisfies Γ and neither satisfies any assertion in Δ nor satisfies $(\exists R.C)(t)$. Therefore, $\forall e \in \Delta^{\mathcal{I}}\ t \notin R^{\mathcal{I}}(t^{\mathcal{I}}, e)$ or $t \notin C^{\mathcal{I}}(e)$. In particular, for $t'^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ we have: if $t \notin R^{\mathcal{I}}(t^{\mathcal{I}}, t'^{\mathcal{I}})$ then $\mathcal{I}$ falsifies $\Gamma \rightarrow \Delta, (\exists R.C)(t), R(t,t')$; if $t \notin C^{\mathcal{I}}(t'^{\mathcal{I}})$ then $\mathcal{I}$ falsifies
$\Gamma \rightarrow \Delta, (\exists R.C)(t), C(t')$.

$\Leftarrow$) Suppose that $\mathcal{I}$ falsifies at least one of the premises. Therefore, $\mathcal{I}$ neither satisfies any assertion in Δ nor satisfies $(\exists R.C)(t)$. It follows that $\mathcal{I}$ falsifies the conclusion $\Gamma \rightarrow \Delta, (\exists R.C)(t)$ of the rule.

case rule $(mpr \rightarrow)$**:**

$\Rightarrow$) Suppose that $\mathcal{I}$ falsifies the conclusion $(\forall R.C)(t), R(t,t'), \Gamma \rightarrow \Delta$. Therefore, $\mathcal{I}$ satisfies $(\forall R.C)(t)$ and $R(t,t')$. By definition of $\mathcal{I}$ it follows that $t \in C^{\mathcal{I}}(t'^{\mathcal{I}})$ and thus, $\mathcal{I}$ satisfies $C(t')$. It follows that $\mathcal{I}$ falsifies the premise
$(\forall R.C)(t), R(t,t'), C(t'), \Gamma \rightarrow \Delta$ of the rule.

$\Leftarrow$) Suppose that $\mathcal{I}$ falsifies the premise $(\forall R.C)(t), R(t,t'), C(t'), \Gamma \rightarrow \Delta$. Then trivially $\mathcal{I}$ falsifies the conclusion $(\forall R.C)(t), R(t,t'), \Gamma \rightarrow \Delta$ of the rule. □

Theorem 1 (Soundness) *If a sequent $\Gamma \rightarrow \Delta$ is provable, then it is valid.*

Proof: If $\Gamma \rightarrow \Delta$ is provable then there is a proof tree T of which it is the conclusion. We use the induction principle applied to the depth n of proof trees.

case n=0: In this case $\Gamma \rightarrow \Delta$ is an axiom. From Lemma 4 it follows that $\Gamma \rightarrow \Delta$ is valid.

induction step: Suppose that for every proof tree of depth n, the conclusion is valid. We will show that still this is true for proof trees of depth $n+1$. Let T be a proof tree of depth $n+1$ with conclusion $\Gamma \to \Delta$.

1. If the conclusion $\Gamma \to \Delta$ is obtained by application of an one-premise rule R, then T is of the form

$$\frac{T'}{\Gamma \to \Delta}$$

where the conclusion of the proof tree T', the sequent S, is the premise of the rule R and $\Gamma \to \Delta$ is the conclusion of the rule R: *i.e.* rule R is of type

$$R\ \frac{S}{\Gamma \to \Delta}$$

Since T' is a proof tree of depth n, by induction it follows that the sequent S is valid. Therefore from Lemma 5 it follows that $\Gamma \to \Delta$ is valid.

2. If the conclusion $\Gamma \to \Delta$ is obtained by application of a two-premises rule R then T is of the form

$$\frac{T' \quad T''}{\Gamma \to \Delta}$$

where the conclusions of the proof trees T' and T'', respectively the sequents S' and S'', are the premises of the rule R and $\Gamma \to \Delta$ is the conclusion of the rule R: *i.e.* rule R is of type

$$R\ \frac{S' \quad S''}{\Gamma \to \Delta}$$

Since T' and T'' are proof trees of depth less or equal than n, by induction it follows that the sequents S' and S'' are valid. Therefore from Lemma 5 it follows that $\Gamma \to \Delta$ is valid. □

Lemma 6 *Let $\mathcal{I}$ be an interpretation. Then,*

1. *for any signed assertion α of type a, $\mathcal{I}$ satisfies α iff $\mathcal{I}$ satisfies both α_1 and α_2;*
2. *for any signed assertion β of type b, $\mathcal{I}$ satisfies β iff $\mathcal{I}$ satisfies β_1 or β_2;*
3. *for any signed assertion γ of type c, $\mathcal{I}$ satisfies γ iff if $\mathcal{I}$ satisfies γ_1 then $\mathcal{I}$ satisfies γ_2, for every $\mathbf{d}$ such that $d \in \Delta^{\mathcal{I}}$;*
4. *for any signed assertion δ of type d, $\mathcal{I}$ satisfies δ iff $\mathcal{I}$ satisfies δ_1 and δ_2, for at least one $\mathbf{d}$ such that $d \in \Delta^{\mathcal{I}}$.*

Proof: The proofs for signed assertions of type a and type b are straightforward.

Case type c assertions: We show only the case of γ in universal quantification form. The other case is similar.

$\Rightarrow$) Suppose that $\mathcal{I}$ satisfies $\gamma = T((\forall R.C)(t))$. By definition, $\mathcal{I}$ satisfies $(\forall R.C)(t)$. It then follows that, for every $d \in \Delta^{\mathcal{I}}$, if $t \in R^{\mathcal{I}}(t^{\mathcal{I}}, \mathbf{d}^{\mathcal{I}})$ then $t \in C^{\mathcal{I}}(\mathbf{d}^{\mathcal{I}})$. Therefore, if $\mathcal{I}$ satisfies $T(R(t, \mathbf{d}))$ then $\mathcal{I}$ satisfies $T(C(\mathbf{d}))$, for every $\mathbf{d}$ such that $d \in \Delta^{\mathcal{I}}$.

$\Leftarrow$) Suppose that if $\mathcal{I}$ satisfies γ_1 then $\mathcal{I}$ satisfies γ_2, for every $\mathbf{d}$ such that $d \in \Delta^{\mathcal{I}}$. Therefore, for every $d \in \Delta^{\mathcal{I}}$, if $t \in R^{\mathcal{I}}(t^{\mathcal{I}}, d)$ then $t \in C^{\mathcal{I}}(d)$. It follows that $\mathcal{I}$ satisfies $(\forall R.C)(t)$ and, thus, by definition $\mathcal{I}$ satisfies $\gamma = T((\forall R.C)(t))$.

Case type d assertions: We show only the case of γ in existential quantification form. The other case is similar.

$\Rightarrow$) Suppose that $\mathcal{I}$ satisfies $\gamma = T((\exists R.C)(t))$. By definition, $\mathcal{I}$ satisfies $(\exists R.C)(t)$. It then follows that there is a $d \in \Delta^{\mathcal{I}}$ such that $t \in R^{\mathcal{I}}(t^{\mathcal{I}}, \mathbf{d}^{\mathcal{I}})$ and $t \in C^{\mathcal{I}}(\mathbf{d}^{\mathcal{I}})$. Therefore, $\mathcal{I}$ satisfies both $T(R(t, \mathbf{d}))$ and $T(C(\mathbf{d}))$.

$\Leftarrow$) Straightforward. □

Lemma 7 *Every Hintikka set S wrt a set of terms H is satisfiable in an interpretation $\mathcal{I}$ with domain H.*

Proof: The interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is defined as follows.

1. The domain $\Delta^{\mathcal{I}}$ is H;
2. every term t is interpreted as the term t, *i.e.* $t^{\mathcal{I}} = t$;
3. For every primitive assertion and negated primitive assertion, for all terms $t, t' \in H$,

$$\begin{array}{rll} t \in A^{\mathcal{I}}(t) & \text{iff} & T(A(t)) \in S \\ f \in A^{\mathcal{I}}(t) & \text{iff} & T((\neg A)(t)) \in S \\ t \in P^{\mathcal{I}}(t, t') & \text{iff} & T(P(t, t')) \in S \end{array}$$

By condition H0, it is easy to see that $\mathcal{I}$ is an interpretation.

We now prove using the induction principle for assertions that $\mathcal{I}$ satisfies α for every signed assertion $\alpha \in S$.

Assume that $T(\alpha) \in S$, where α is a primitive assertion or a negated primitive assertion. Then by definition, $\mathcal{I}$ satisfies $T(\alpha)$.

Similarly, if $NT(\alpha) \in S$, where α is a primitive assertion or a negated primitive assertion, then $\mathcal{I}$ satisfies $NT(\alpha)$.

If a type-a signed assertion α is in S, then by condition $H1$, both α_1 and α_2 are in S. By induction, $\mathcal{I}$ satisfies both α_1 and α_2. Therefore, by Lemma 6, $\mathcal{I}$ satisfies α.

The case of type-b signed assertion β is similar.

If a signed assertion γ of type c is in S, then by condition $H3$, for every term $t' \in \Delta^{\mathcal{I}}$, if $\gamma_1 \in S$ then $\gamma_2 \in S$. By induction and since $t'^{\mathcal{I}} = t'$, for every $\mathbf{d}$ such that $d \in \Delta^{\mathcal{I}}$ (where $d = t'^{\mathcal{I}}$), if $\mathcal{I}$ satisfies γ_1 then $\mathcal{I}$ satisfies γ_2. From Lemma 6 it follows that $\mathcal{I}$ satisfies γ.

If a type-d assertion δ is in S, then by condition $H4$ there is at least on term $t' \in H$ such that both δ_1 and δ_2 are in S. By induction and since $t'^{\mathcal{I}} = t'$, for at least one $\mathbf{d}$ such that $d \in \Delta^{\mathcal{I}}$ (where $d = t'^{\mathcal{I}}$), $\mathcal{I}$ satisfies both δ_1 and δ_2. From Lemma 6 it follows that $\mathcal{I}$ satisfies δ. □

Lemma 9 *The* **Search** *procedure satisfies the following conditions:*

1. *If the input sequent $\Gamma \rightarrow \Delta$ is valid, then the procedure* **Search** *halts with a finite closed tree T which is a proof tree for $\Gamma \rightarrow \Delta$.*

2. *If the input sequent $\Gamma \rightarrow \Delta$ is falsifiable, either* **Search** *halts with a finite counterexample tree T and $\Gamma \rightarrow \Delta$ is falsifiable in an interpretation with finite domain, or* **Search** *generates an infinite tree T and $\Gamma \rightarrow \Delta$ is falsifiable in an interpretation with a countably infinite domain.*

Proof: First, assume that the sequent $\Gamma \rightarrow \Delta$ is falsifiable. If the tree T was finite and each leaf is an axiom, by Theorem 1, $\Gamma \rightarrow \Delta$ would be valid, a contradiction. Hence, either T is finite and contains some path to a non axiom leaf, or T is infinite and by König's lemma contains an infinite path. In either case, we show that a Hintikka set can be found along that path. Let U be the union of all assertions occurring in the left-hand side of each sequent along that path and V be the union of all assertions occurring in the right-hand side of any such sequent. Let

$$S = \{T(\alpha) | \alpha \in U\} \cup \{NT(\beta) | \beta \in V\}$$

We prove the following claim.

Claim 1 *S is a Hintikka set wrt the set of terms consisting of the set H of terms in $Terms_{Used}$.*

Proof:

1. H0 holds. Since every primitive or negated primitive assertion occurring in a sequent occurs in every path having this sequent as source, if S contains a conjugated pair then some sequent in the path is an axiom. This contradicts the fact that either the path is finite and ends in a non axiom, or is an infinite path;

2. H1 and H2 hold. This is true because the definition of *a-* and *b-components* mirrors the inference rules:

   (a) for every assertion $\alpha \in U$, if α belongs to $\Gamma \rightarrow \Delta$ and $T(\alpha)$ is of type a, then α_1 and α_2 are added to the successor of $\Gamma \rightarrow \Delta$, and thus are in S;

   (b) for every assertion $\beta \in U$, if β belongs to $\Gamma \rightarrow \Delta$ and $T(\beta)$ is of type b, then β_1 is added to the left successor of $\Gamma \rightarrow \Delta$ and β_2 is added to the right successor of $\Gamma \rightarrow \Delta$, during the expansion step. Hence, either β_1 or β_2 belongs to S.

   The same holds for the set V;

3. H3 holds. Every time an assertion γ, such that its signed assertion is of type c, is expanded, then for all terms in $Terms_{Used}$ that have not already been used with γ, if $\gamma_1 \in S$ then $\gamma_2 \in S$. Hence, H3 holds;

4. H4 holds. Every time an assertion δ, such that its signed assertion is of type d, is expanded, x is added to $Terms_{Used}$ and the substitution instance is added to the upper sequent. Hence, H4 is satisfied and the claim holds. □

By Lemma 7 there is an interpretation $\mathcal{I}$ which satisfies S. This implies that $\mathcal{I}$ falsifies $\Gamma \rightarrow \Delta$.

Note that H must be infinite if the tree T is infinite. Otherwise, since **Search** starts with a finite sequent, every path would be finite and would end either with an axiom or a finished sequent.

If the sequent $\Gamma \rightarrow \Delta$ is valid then the tree T must be a proof tree since otherwise, the above argument shows that $\Gamma \rightarrow \Delta$ is falsifiable. □

B.2 Proofs of Section 4.5

Theorem 3 (Four-valued interpolation theorem) $\Gamma \rightarrow \Delta$ *is a valid sequent iff there exists a constructible interpolant* γ *of* $\Gamma \rightarrow \Delta$.

Proof: The *(if)* direction is straightforward: if γ is an interpolant of $\Gamma \rightarrow \Delta$, then $\Gamma \rightarrow \Delta$ is valid.

(Only if) direction. If $\Gamma \rightarrow \Delta$ is valid then there is a proof tree T of which it is the conclusion. We use the induction principle applied to the depth n of proof trees.

case n=0: In this case $\Gamma \rightarrow \Delta$ is an axiom of the form $\alpha, \Gamma' \rightarrow \alpha, \Delta'$. $\gamma = \alpha$ is an interpolant of $\Gamma \rightarrow \Delta$.

induction step: Suppose that for every proof tree of depth n, the conclusion is valid. We will show that still this is true for proof trees of depth $n+1$. Let T be a proof tree of depth $n+1$ with conclusion $\Gamma \to \Delta$. Then T has one of the following forms:

form 1: T is of the form

$$\frac{T'}{\Gamma \to \Delta}$$

if the conclusion $\Gamma \to \Delta$ is obtained by application of an one-premise rule R, where the conclusion of the proof tree T', the sequent S, is the premise of the rule R and $\Gamma \to \Delta$ is the conclusion of the rule R;

form 2: T is of the form

$$\frac{T' \quad T''}{\Gamma \to \Delta}$$

if the conclusion $\Gamma \to \Delta$ is obtained by application of a two-premises rule R, where the conclusions of the proof trees T' and T'', respectively the sequents S' and S'', are the premises of the rule R and $\Gamma \to \Delta$ is the conclusion of the rule R.

We proceed by case analysis on the rules of Definition 11.

case rule $(\sqcap \to)$**:** In this case T is of form 1, $\Gamma \to \Delta$ is of form $(C \sqcap D)(t), \Gamma \to \Delta$ and the sequent S is of form $C(t), D(t), \Gamma' \to \Delta$. By induction on tree T', there exists an interpolant γ of S. Therefore, γ is an interpolant of $\Gamma \to \Delta$.

case rule $(\to \sqcap)$**:** In this case T is of form 2, $\Gamma \to \Delta$ is of form $\Gamma \to \Delta, (C \sqcap D)(t)$, S' is of form $\Gamma \to \Delta, C(t)$ and S'' is of form $\Gamma \to \Delta, D(t)$. By induction on T' and T'', there exist interpolant γ' of S' and γ'' of S''. Let γ be $\gamma' \wedge \gamma''$. γ is an interpolant of $\Gamma \to \Delta$.

case rule $(\sqcup \to)$**:** In this case T is of form 2, $\Gamma \to \Delta$ is of form $(C \sqcup D)(t), \Gamma \to \Delta$, S' is of form $C(t), \Gamma' \to \Delta$ and S'' is of form $D(t), \Gamma' \to \Delta$. By induction on T' and T'', there exist interpolant γ' of S' and γ'' of S''. Let γ be $\gamma' \vee \gamma''$. γ is an interpolant of $\Gamma \to \Delta$.

case rule $(\to \sqcup)$**:** In this case T is of form 1, $\Gamma \to \Delta$ is of form $\Gamma \to \Delta, (C \sqcup D)(t)$ and S is of form $\Gamma \to \Delta, C(t), D(t)$. By induction on T' there exists an interpolant γ of S. Therefore, γ is an interpolant of $\Gamma \to \Delta$.

case rule $(\to \forall)$**:** In this case T is of form 1, $\Gamma \to \Delta$ is of form $\Gamma \to \Delta, (\forall R.C)(t)$ and S is of form $\Gamma, R(t,x) \to \Delta, C(x)$. By induction on T' there exists an interpolant γ' of S. Let γ be following transformation of γ':

1. γ' can be rewritten as $(\gamma_1\ op'_1\ \gamma'_1)\ op_1\ (\gamma_2\ op'_2\ \gamma'_2)\ op_2\ \ldots\ (\gamma_n\ op'_n\ \gamma'_n)$ such that $n \geq 1$ and (i) in every assertion occurring in γ_i, x occurs, (ii) γ'_i can not be empty (for $i < n$) and x does not occur in γ'_i and (iii) $op_i, op'_i \in \{\wedge, \vee\}$;

2. for each γ_i, remove occurrences of assertion $R(t,x)$;
3. for each γ_i, replace the $\wedge$ and $\vee$ operators in γ_i with the $\sqcap$ and $\sqcup$ operators, respectively;
4. for each γ_i, replace all assertions $C(x)$ occurring in γ_i with C and let γ_i^c be the obtained concept;
5. for each γ_i^c, replace it with $\forall R.\gamma_i^c$.

γ is an interpolant of $\Gamma \rightarrow \Delta$.

case rule $(\exists \rightarrow)$**:** In this case T is of form 1, $\Gamma \rightarrow \Delta$ is of form $(\exists R.C)(t), \Gamma \rightarrow \Delta$ and S is of form $R(t,x), C(x), \Gamma \rightarrow \Delta$. By induction on T' there exists an interpolant γ' of S. Let γ be following transformation of γ':

1. γ' can be rewritten as $(\gamma_1\ op'_1\ \gamma'_1)\ op_1\ (\gamma_2\ op'_2\ \gamma'_2)\ op_2\ \ldots\ (\gamma_n\ op'_n\ \gamma'_n)$ such that $n \geq 1$ and (i) in every assertion occurring in γ_i, x occurs, (ii) γ'_i can not be empty (for $i < n$) and x does not occur in γ'_i and (iii) $op_i, op'_i \in \{\wedge, \vee\}$;
2. for each γ_i, remove occurrences of assertion $R(t,x)$;
3. for each γ_i, replace the $\wedge$ and $\vee$ operators in γ_i with the $\sqcap$ and $\sqcup$ operators, respectively;
4. for each γ_i, replace all assertions $C(x)$ occurring in γ_i with C and let γ_i^c be the obtained concept;
5. for each γ_i^c, replace it with $\exists R.\gamma_i^c$.

γ is an interpolant of $\Gamma \rightarrow \Delta$.

case rule $(\rightarrow \exists)$**:** In this case T is of form 2, $\Gamma \rightarrow \Delta$ is of form $\Gamma \rightarrow \Delta, (\exists R.C)(t)$, S' is of form $\Gamma \rightarrow \Delta, (\exists R.C)(t), R(t,t')$ and S'' is of form $\Gamma \rightarrow \Delta, (\exists R.C)(t), C(t')$. By induction on T' and T'', there exist interpolant γ' of S' and γ'' of S''. Let γ be $\gamma' \wedge \gamma''$. γ is an interpolant of $\Gamma \rightarrow \Delta$.

case rule $(mpr \rightarrow)$**:** In this case T is of form 1, $\Gamma \rightarrow \Delta$ is of form $(\forall R.C)(t), R(t,t'), \Gamma \rightarrow \Delta$ and S is of form $(\forall R.C)(t), R(t,t'), C(t'), \Gamma \rightarrow \Delta$. By induction on T' there exists an interpolant γ of S. Therefore, γ is an interpolant of $\Gamma \rightarrow \Delta$. $\square$

B.3 Proofs of Section 4.6

Theorem 4 *The problem of determining the validity of a sequent* $\Gamma \rightarrow \Delta$ *is co-NP-Hard.*

Proof: The proof consist in a reduction of the propositional tautological entailment problem. Let α and β be two propositional formulae. The problem of determining whether $\alpha \models_4 \beta$, in standard four-valued semantics, is a co-NP-Hard problem [Patel-Schneider, 1987a], where $\models_4$ is the standard four-valued entailment relation.

Let γ be a propositional formula. Let γ^c be the concept obtained from γ by replacing in γ the operators for conjunction, disjunction and negation with the operators $\sqcap$, $\sqcup$ and $\neg$, respectively. It follows that $\alpha \models_4 \beta$ iff $\alpha^c(a) \rightarrow \beta^c(a)$ is valid, where a is an individual. □

Theorem 5 *Determining the validity of a sequent $\Gamma \rightarrow \Delta$ is decidable.* ∎

We will give only a sketch of the decidability proof. The detailed proof can be easily derived from this sketch.

First of all, we introduce the notion of *generation.*

Definition 43 *Let x be a variable and t a term. Then t* generates *x, written $t \mapsto x$, if and only if x is the new variable introduced by the application of the $(\rightarrow \forall)$ rule to $(\forall R.C)(t)$.* ∎

Note, there is a infinite counterexample tree T with conclusion $\Gamma \rightarrow \Delta$, if and only if there is an infinite path p from the root, such that there is an infinite chain $t \mapsto x_1 \mapsto \ldots \mapsto x_n \mapsto \ldots$ of generated variables along that path p. Therefore, it is sufficient to "intercept" these infinitary paths in order to get decidability

Definition 44 *Let $\Gamma \rightarrow \Delta$ be a sequent such that $(\forall R.C)(x)$ appears in Δ. A node labelled $\Gamma \rightarrow \Delta$ is called* ∞-stopped *if and only if, if $t \mapsto x$ then both*

1. *$R(t, x)$ can not be involved in an applicable left hand side rule $(R \rightarrow)$;*
2. *no right hand side rule $(\rightarrow R)$ is applicable to assertions different from $(\forall R.C)(x)$.* ∎

Definition 45 *A node is* ∞-finished *if and only if it is finished or is ∞-stopped.* ∎

Definition 46 *An* ∞-counterexample tree *is a deduction tree where each leaf is ∞-finished and there is at least one non axiom leaf.* ∎

Now we sketch out the main lemma.

Lemma 17 *Every ∞-counterexample tree with conclusion $\Gamma \rightarrow \Delta$ can be expanded to a counterexample tree with conclusion $\Gamma \rightarrow \Delta$.*

Proof: Let T_∞ be an ∞-counterexample tree with conclusion $\Gamma \rightarrow \Delta$. For each non axiom leaf L_i (there is at least one and finitely many) labelled $\Gamma_{L_i} \rightarrow \Delta_{L_i}$ we know that: if $t \mapsto x$ then

1. no $(R \rightarrow)$ rule is applicable;
2. no $(\rightarrow R)$ rule different from $(\rightarrow \forall)$ is applicable.

Let T be the deduction tree obtained from T_∞ by expanding completely nodes L_i by the usual rules (thus, T could be infinite).

Let T_{L_i} be the subtree of T such that the root of T_{L_i} is L_i. We will show that T_{L_i} is a counterexample tree for $\Gamma_{L_i} \rightarrow \Delta_{L_i}$ and , therefore, T is a counterexample tree for $\Gamma \rightarrow \Delta$.

If the depth of T_{L_i} is 0, then T_{L_i} is a one node tree labelled $\Gamma_{L_i} \rightarrow \Delta_{L_i}$. Since $\Gamma_{L_i} \rightarrow \Delta_{L_i}$ is not an axiom and no rule is applicable to it, it follows that $\Gamma_{L_i} \rightarrow \Delta_{L_i}$ is falsifiable and, thus, T_{L_i} is a counterexample tree for $\Gamma_{L_i} \rightarrow \Delta_{L_i}$.

Otherwise, a $(\rightarrow \forall)$ is applied to a $(\forall R.C)(x) \in \Delta_{L_i}$ and there is a term t such that $t \mapsto x$, and there is a $R'(t,x) \in \Gamma_{L_i}$ and no $R''(t,x) \in \Gamma_{L_i}$ is usable in a $(R \rightarrow)$ rule. Therefore, we have that T_{L_i} is the tree

$$\dfrac{\dfrac{S'_{L_i}}{\Gamma'_{L_i}, R'(t,x), R(x,y) \rightarrow \Delta'_{L_i}, C(y)}}{\Gamma'_{L_i}, R'(t,x) \rightarrow \Delta'_{L_i}, (\forall R.C)(x)}$$

Let T'_{L_i} be the subtree of T_{L_i} such that the root of T'_{L_i} is labelled S'_{L_i}. Let S_1 be the sequent $\Gamma'_{L_i}, R'(t,x) \rightarrow \Delta'_{L_i}, (\forall R.C)(x)$ (which is the sequent labelled by L_i) and let S_2 be the sequent $\Gamma''_{L_i}, R'(t,x), R(x,y) \rightarrow \Delta'_{L_i}, C(y)$. We have

1. no left rule $(R \rightarrow)$ is applicable to S_1;
2. no left rule $(R \rightarrow)$ is applicable to S_2;
3. $\{\Gamma'_{L_i}, R'(t,x)\} \cap \{\Delta'_{L_i}, (\forall R.C)(x)\} = \emptyset$;
4. since y is a new variable, $\{\Gamma'_{L_i}, R'(t,x), R(x,y)\} \cap \{\Delta'_{L_i}, (\forall R.C)(x), C(y)\} = \emptyset$;
5. since no $R''(t,x) \in \Gamma_{L_i}$ is usable in a $(R \rightarrow)$ rule, no $C(y)$ will be generated on the left hand side along an infinitary path of T'_{L_i}.

Therefore, there is a possible infinite path from T'_{L_i} from which a Hintikka set can be constructed, as for Lemma 9, such that T'_{L_i} is a counterexample tree for S'_{L_i} and, thus, T_{L_i} is a counterexample tree for $\Gamma_{L_i} \rightarrow \Delta_{L_i}$. $\square$

Therefore,

Lemma 18 *If the input sequent $\Gamma \rightarrow \Delta$ is falsifiable,* **Search** *halts with a finite ∞-counterexample tree T and $\Gamma \rightarrow \Delta$ is falsifiable in an interpretation with possible infinite domain.*

Proof: First, note that **Search** halts since no infinite path p from the root, such that there is an infinite chain $t \mapsto x_1 \mapsto \ldots \mapsto x_n \mapsto \ldots$ of generated variables along that path p, is generated by **Search**.

Now assume that the sequent $\Gamma \rightarrow \Delta$ is falsifiable. If the tree T was finite and each leaf is an axiom, by Theorem 1, $\Gamma \rightarrow \Delta$ would be valid, a contradiction. Hence, T is finite and contains some path to a non axiom ∞-finished leaf. From Lemma 17 follows that T is a counterexample tree for $\Gamma \rightarrow \Delta$ from which an Hintikka set can be build as for Lemma 9 falsifying $\Gamma \rightarrow \Delta$.

If the sequent $\Gamma \rightarrow \Delta$ is valid then the tree T must be a proof tree since otherwise, the above argument shows that $\Gamma \rightarrow \Delta$ is falsifiable. $\Box$

B.4 Proofs of Section 5.2

In the following, since disjunction can be expressed in terms of negation and conjunction, and universal quantification on roles ca be expressed in terms of negation and existential quantification on roles, whenever possible we will leave out case proofs involving disjunction and universal quantification.

In the following, let (Σ, Ω) be an MIRLOG knowledge base, let $\mathcal{I}$ be a model of (Σ, Ω), R a role, a, b individuals and a closed. With $R^{+}_{\Sigma}(a)$, $R^{-}_{\Sigma}(a)$, and $R^{\mathcal{I}}_{\Sigma}(a, x)$ (for $x \in \{t, f\}$), we indicate respectively the sets

$$\begin{array}{lcl} R^{+}_{\Sigma}(a) & = & \{\gamma(b) : \Sigma \not\approx R(a,b)\} \\ R^{-}_{\Sigma}(a) & = & \{\gamma(b) : \Sigma \not\not\approx R(a,b)\} \\ R^{\mathcal{I}}_{\Sigma}(a,t) & = & \{p \in \Delta : t \in R^{\mathcal{I}}(\gamma(a), p)\} \\ R^{\mathcal{I}}_{\Sigma}(a,f) & = & \{p \in \Delta : f \in R^{\mathcal{I}}(\gamma(a), p)\} \end{array}$$

By definition of model $\mathcal{I}$, since a is closed it follows easily that

$$\begin{array}{lcl} R^{+}_{\Sigma}(a) & = & R^{\mathcal{I}}_{\Sigma}(a,t) \\ R^{-}_{\Sigma}(a) & = & R^{\mathcal{I}}_{\Sigma}(a,f) \end{array}$$

Moreover, since a is closed it happens that for each primitive concept A

$$A^{\mathcal{I}}(\gamma(a) = \{t\} \text{ or } A^{\mathcal{I}}(\gamma(a) = \{f\}$$

Similar for roles $R(a, b)$.

As last, it can be easily verified that if b does not appear in Σ then $\Sigma \not\approx R(a, b)$.

Propositions 6 to 11 follow from the following lemmas.

Lemma 19 *Let (Σ, Ω) be a KB, $\texttt{Cl}(a) \in \Omega$ and α an assertion $C(a)$ or $R(a, b)$, such that $\mathcal{I}$ is a model of (Σ, Ω). Then the following hold:*

1. *$\mathcal{I}$ satisfies α or $\mathcal{I}$ f-satisfies α, for any quantifier free C and for any R;*
2. *if (Σ, Ω) is completely closed, then $\mathcal{I}$ satisfies α or $\mathcal{I}$ f-satisfies α, for any C and R.*

Proof: The proof is given on induction on the structure of α. Let $\mathcal{I}$ be a model of (Σ, Ω).

case $\alpha = A(a)$ **:** If $\mathcal{I}$ satisfies $A(a)$ then we are done. Suppose $\mathcal{I}$ does not satisfy α. Then there exists a model of Σ such that $t \notin A^{\mathcal{I}}(\gamma(a))$. By definition, it follows that $f \in A^{\mathcal{I}}(\gamma(a))$. Hence $\mathcal{I}$ f-satisfies $A(a)$.

case $\alpha = P(a, b)$ **:** Similar as above.

case $\alpha = \neg C(a)$ **:** If $\mathcal{I}$ satisfies α then we are done. Suppose $\mathcal{I}$ does not satisfy α. Hence, $\mathcal{I}$ does not f-satisfy $C(a)$. By induction on C, it follows that $\mathcal{I}$ satisfies $C(a)$, hence $\mathcal{I}$ f-satisfies $\neg C(a)$.

case $\alpha = \neg P(a, b)$ **:** Similar as above.

case $\alpha = (C \sqcap D)(a)$ **:** Straightforward.

This completes the first part.

case $\alpha = (\exists R.C)(a)$ **:** If $\mathcal{I}$ satisfies α then we are done. Suppose $\mathcal{I}$ does not satisfy α. Since a is closed it follows that $R^{+}_{\Sigma}(a) = R^{\mathcal{I}}_{\Sigma}(a, t)$. Let b be an individual such that $\gamma(b) \in R^{+}_{\Sigma}(a)$. Hence, b appears in Σ and $t \notin C^{\mathcal{I}}(\gamma(b))$. Since b is closed, by induction on C, it follows that $\mathcal{I}$ f-satisfies $C(b)$. Hence, $\mathcal{I}$ f-satisfies α. □

Lemma 20 *Let (Σ, Ω) be a KB, $\texttt{Cl}(a) \in \Omega$ and α an assertion $C(a)$ or $R(a, b)$, such that $\mathcal{I}$ is a model of (Σ, Ω). Then the following hold:*

1. *$\mathcal{I}$ satisfies (f-satisfies) α if and only if all models of (Σ, Ω) satisfy (f-satisfy) α, for any quantifier free C and R;*

2. *if (Σ, Ω) is completely closed, then $\mathcal{I}$ satisfies (f-satisfies) α if and only if all models of (Σ, Ω) satisfy (f-satisfy) α, for any C and R.*

Proof:

$\Leftarrow$) Trivial.

$\Rightarrow$) The proof is given on induction on the structure of α. Let $\mathcal{I}$ be a model of (Σ, Ω).

case $\alpha = A(a)$ **:** Since $\mathcal{I}$ satisfies $A(a)$ it follows that $t \in A^{\mathcal{I}}(\gamma(a))$. Since a is closed, by definition, $t \in A^{\mathcal{J}}(\gamma(a))$ for all models $\mathcal{J}$ of Σ and, thus, for all models of (Σ, Ω). Similar for f-satisfiability.

case $\alpha = P(a, b)$ **:** Similar as above.

case $\alpha = \neg C(a)$ **:** Since $\mathcal{I}$ satisfies $\neg C(a)$, it follows that $\mathcal{I}$ f-satisfies $C(a)$. Since a is closed, by induction it follows that all models of (Σ, Ω) f-satisfy $C(a)$. Hence, all models of (Σ, Ω) satisfy $\neg C(a)$. Similar for f-satisfaction.

case $\alpha = \neg P(a, b)$ **:** Similar as above.

case $\alpha = (C \sqcap D)(a)$ **:** Straightforward.

This completes the first part of the proposition.

case $\alpha = (\exists R.C)(a)$ **:** Since $\mathcal{I}$ satisfies α and since a is closed, it follows that $|R^{+}_{\Sigma}(a)| \geq 1$ and $t \in C^{\mathcal{I}}(\gamma(b))$, for some $\gamma(b) \in R^{+}_{\Sigma}(a)$. Since a is closed it follows that $R^{+}_{\Sigma}(a) = R^{\mathcal{I}}_{\Sigma}(a, t)$ and, thus, b appears in Σ. Hence b is closed. Therefore, by induction on R and C, for all models $\mathcal{J}$ of (Σ, Ω), $t \in C^{\mathcal{J}}(\gamma(b))$. Hence, all models of (Σ, Ω) satisfy α. $\square$

B.5 Proofs of Section 5.3

Lemma 12 *No axiom is falsifiable. Equivalently, every axiom is valid.*

Proof: The proof for $\alpha, \Gamma \rightarrow_{(\Sigma,\Omega)} \alpha, \Delta$, $\Gamma \rightarrow_{(\Sigma,\Omega)} \top(o), \Delta$ and $\bot(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is straightforward.

Let $\mathcal{I}$ be an interpretations which (Σ, Ω)-satisfies Γ. We show that if $\mathtt{Cl}(a) \in \Omega$ and $\Sigma \not\approx A(a)$, then $\Gamma \rightarrow_{(\Sigma,\Omega)} \neg A(a), \Delta$ is valid. Since a is closed and $(\mathcal{I}, \mathcal{M}(\Sigma))$ satisfies $\mathtt{Cl}(a)$, it follows that $A^{\mathcal{I}}(\gamma(a)) = \{f\}$. Therefore, $\mathcal{I}$ satisfies $\neg A(a)$.

Finally, the proof for axiom $\Gamma \rightarrow_{(\Sigma,\Omega)} \neg P(a, b), \Delta$ is similar. $\square$

Lemma 13 *For each of the rules in Definition 35, the conclusion of a rule is falsifiable iff at least one of the premises of the rule is falsifiable. Equivalently, the conclusion of a rule is valid iff all premises of the rule are valid.*

Proof: The proof consists in a case analysis on the rules of Definition 35. Remember that an interpretation $\mathcal{I}$ falsifies a sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ iff $\mathcal{I}$ (Σ,Ω)-satisfies Γ and $\mathcal{I}$ does not satisfy any of the assertional formulae in Δ.

We will give only a proof for significant cases.

case rule $(\bot_{C1}\rightarrow)$**:** Suppose that $\mathtt{Cl}(a) \in \Omega$ and $\Sigma \not\models\!\!\approx A(a)$. Therefore, every epistemic model $(\mathcal{I}, \mathcal{M}(\Sigma))$ satisfying $\mathtt{Cl}(a)$ is such that $A^{\mathcal{I}}(\gamma(a)) = \{f\}$. Hence, there could no interpretation satisfying $A(a)$. Therefore, $A(a), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid. Moreover, $\bot(a), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid.

case rule $(\bot_{C2}\rightarrow)$**:** Straightforward, observing that for a closed individual a, $A^{\mathcal{I}}(\gamma(a)) = \{f\}$ or $A^{\mathcal{I}}(\gamma(a)) = \{t\}$.

case rule $(\rightarrow \forall)$**:** The case $\mathtt{Cl}(o) \notin \Omega$ is as for standard four-valued semantics.

Suppose $\mathtt{Cl}(o) \in \Omega$. The case $\delta = \top(a)$ is straightforward. In fact, there are no role fillers for $R(a)$, hence $(\forall R.C)(a) \equiv \top(a)$.

Let us consider the case δ is $C(a_1) \wedge \ldots \wedge C(a_n)$, where $a_i \in \mathcal{O}$, with $(i > 0)$, are all the individuals such that $\Sigma \models\!\!\approx P(a, a_i)$ (if $R = P$), else $\Sigma \not\models\!\!\approx P(a, a_i)$

$\Rightarrow$) Suppose that $\mathcal{I}$ falsifies the conclusion $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, (\forall R.C)(a)$. Therefore, $\mathcal{I}$ (Σ,Ω)-satisfies Γ, does not satisfy any assertional formula in Δ and does not satisfy $(\forall R.C)(a)$. Since a is closed there is an individual a_i such that $\Sigma \models\!\!\approx P(a, a_i)$ (if $R = P$), else $\Sigma \not\models\!\!\approx P(a, a_i)$) and $\mathcal{I}$ does not satisfy $C(a_i)$. Therefore, $\mathcal{I}$ falsifies $\Gamma', \rightarrow_{(\Sigma,\Omega)} \Delta, \delta$.

$\Leftarrow$) Straightforward.

case rule $(\exists \rightarrow)$**:** The case $\mathtt{Cl}(o) \notin \Omega$ is as for standard four-valued semantics.

Suppose $\mathtt{Cl}(o) \in \Omega$. The case $\gamma = \bot(a)$ is straightforward. In fact, in this case there are no role fillers for $R(a)$, hence $(\exists R.C)(a) \equiv \bot(a)$.

Let us consider the case γ is $(R(o, a_1) \wedge C(a_1)) \vee \ldots \vee (R(o, a_n) \wedge C(a_n))$, where $a_i \in \mathcal{O}$, with $(i > 0)$, are all the individuals such that $\Sigma \models\!\!\approx P(a, a_i)$ (if $R = P$), else $\Sigma \not\models\!\!\approx P(a, a_i)$). Suppose that $\mathcal{I}$ (Σ,Ω)-satisfies Γ. It easy to see that $\mathcal{I}$ satisfies γ iff $\mathcal{I}$ satisfies $(\exists R.C)(a)$. Hence, the conclusion of the rule is falsifiable iff the premise is falsifiable.

Theorem 6 (Soundness) *If a sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is provable, then it is valid.*

Proof: If $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is provable then there is a proof tree T of which it is the conclusion. We use the induction principle applied to the depth n of proof trees.

case n=0: In this case $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is an axiom. From Lemma 12 it follows that $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid.

induction step: Suppose that for every proof tree of depth n, the conclusion is valid. We will show that still this is true for proof trees of depth $n+1$. Let T be a proof tree of depth $n+1$ with conclusion $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

1. If the conclusion $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is obtained by application of an one-premise rule R, then T is of the form

$$\frac{T'}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

where the conclusion of the proof tree T', the sequent S, is the premise of the rule R and $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is the conclusion of the rule R: *i.e.* rule R is of type

$$R\ \frac{S}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

Since T' is a proof tree of depth n, by induction it follows that the sequent S is valid. Therefore from Lemma 13 it follows that $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid.

2. If the conclusion $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is obtained by application of a two-premises rule R then T is of the form

$$\frac{T' \quad T''}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

where the conclusions of the proof trees T' and T'', respectively the sequents S' and S'', are the premises of the rule R and $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is the conclusion of the rule R: *i.e.* rule R is of type

$$R\ \frac{S' \quad S''}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

Since T' and T'' are proof trees of depth less or equal than n, by induction it follows that the sequents S' and S'' are valid. Therefore from Lemma 13 it follows that $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid. □

Lemma 14 *Let $\mathcal{I}$ be an interpretation. Then:*

1. *for any signed assertional formula γ^a of type a, $\mathcal{I}$ satisfies γ^a iff $\mathcal{I}$ satisfies both γ^a_1 and γ^a_2;*

2. *for any signed assertion γ^b of type b, $\mathcal{I}$ satisfies γ^b iff $\mathcal{I}$ satisfies γ^b_1 or γ^b_2;*

3. *for any signed assertion* γ^c *of type* c, $\mathcal{I}$ *satisfies* γ^c *iff if* $\mathcal{I}$ *satisfies* γ_1^c *then* $\mathcal{I}$ *satisfies* γ_2^c, *for every* $\mathbf{p}$ *in place of* o';

4. *for any signed assertion* γ^d *of type* d, $\mathcal{I}$ *satisfies* γ^d *iff* $\mathcal{I}$ *satisfies* γ_1^d *and* γ_2^d, *for at least one* $\mathbf{p}$ *in place of* o'.

Proof: The proof is as for the same lemma for standard four-valued semantics. □

Lemma 15 *Every Hintikka set* S *wrt a set of objects* H *is satisfiable.*

Proof: The interpretation $\mathcal{I}$ is defined as follows.

1. $\mathcal{I}$ is injective on objects, *i.e.* $o^{\mathcal{I}} \neq o'^{\mathcal{I}}$, if $o \neq o'$;

2. For every primitive assertion and negated primitive assertion, for all parameters $p, p' \in \Delta$,

$$\begin{array}{rcl} t \in A^{\mathcal{I}}(p) & \text{iff} & T(A(o)) \in S \text{ and } o^{\mathcal{I}} = p \\ f \in A^{\mathcal{I}}(p) & \text{iff} & T((\neg A)(o)) \in S \text{ and } o^{\mathcal{I}} = p \\ t \in P^{\mathcal{I}}(p,p') & \text{iff} & T(P(o,o')) \in S \text{ and } o^{\mathcal{I}} = p, o'^{\mathcal{I}} = p' \\ f \in P^{\mathcal{I}}(p,p') & \text{iff} & T(\neg P(p,p')) \in S \text{ and } o^{\mathcal{I}} = p, o'^{\mathcal{I}} = p' \end{array}$$

Note that neither $T(\bot(a))$ nor $NT(\top(a))$ are in S. By condition H0, it is easy to see that $\mathcal{I}$ is an interpretation.

We now prove using the induction principle for assertional formulae that $\mathcal{I}$ satisfies γ for every signed assertional formulae $\gamma \in S$.

Assume that $T(\gamma) \in S$, where γ is a primitive assertion or a negated primitive assertion. Then by definition, $\mathcal{I}$ satisfies $T(\gamma)$.

Similarly, if $NT(\gamma) \in S$, where γ is a primitive assertion or a negated primitive assertion, then $\mathcal{I}$ satisfies $NT(\gamma)$.

If $T(\top(a)) \in S$ the by definition $\mathcal{I}$ satisfies $\top(a)$. similar for $NT(\bot(a)) \in S$.

If a type-a signed assertion γ^a is in S, then by condition $H1$, both γ_1^a and γ_2^a are in S. By induction, $\mathcal{I}$ satisfies both γ_1^a and γ_2^a. Therefore, by Lemma 14, $\mathcal{I}$ satisfies α.

The case of type-b signed assertion γ^b is similar.

If a type-c assertional formula γ^c is in S, then then by condition $H3$, for every objects $o \in H$, if γ_1^c is in S then γ_2^c is in S. By induction and definition of $\mathcal{I}$ on objects, for every $\mathbf{p}$, if $\mathcal{I}$ satisfies γ_1^c then $\mathcal{I}$ satisfies γ_2^c. From Lemma 14 it follows that $\mathcal{I}$ satisfies γ^c.

Similar for type-d assertional formulae γ^d. □

Lemma 16 *The* **Search** *procedure satisfies the following conditions:*

1. *If the input sequent* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is valid, then the procedure* **Search** *halts with a finite closed tree* T *which is a proof tree for* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.
2. *If the input sequent* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is falsifiable, either* **Search** *halts with a finite counterexample tree* T *and* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is falsifiable, or* **Search** *generates an infinite tree* T *and* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is falsifiable.*

Proof: First, assume that the sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is falsifiable. If the tree T was finite and each leaf is an axiom, by Theorem 6, $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ would be valid, a contradiction. Hence, either T is finite and contains some path to a non axiom leaf, or T is infinite and by König's lemma contains an infinite path. In either case, we show that a Hintikka set can be found along that path. Let U be the union of all assertional formulae occurring in the left-hand side of each sequent along that path and V be the union of all assertional formulae occurring in the right-hand side of any such sequent. Let

$$S = \{T(\gamma)|\gamma \in U\} \cup \{NT(\delta)|\delta \in V\}$$

We prove the following claim.

Claim 2 *S is a Hintikka set wrt the set of objects consisting of the set H of objects in $Terms_{Used}$.*

Proof:

1. H0 holds. Since every primitive or negated primitive assertion occurring in a sequent occurs in every path having this sequent as source, if S contains a conjugated pair then some sequent in the path is an axiom. This contradicts the fact that either the path is finite and ends in a non axiom, or is an infinite path. Similarly, neither $T(\bot(a))$ nor $NT(\top(a))$ are in S.
2. H1 and H2 hold. This is true because the definition of *a-components* and *b-components* mirrors the inference rules:
   (a) for every assertional formulae $\gamma^a \in U$, if γ^a belongs to $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ and $T(\gamma^a)$ is of type a, then γ_1^a and γ_2^a are added to the successor of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$, and thus are in S;
   (b) for every assertional formulae $\gamma^b \in U$, if γ^b belongs to $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ and $T(\gamma^b)$ is of type b, then γ_1^b is added to the left successor of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ and γ_2^b is added to the right successor of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$, during the expansion step. Hence, either γ_1^b or γ_2^b belongs to S.

The same holds for the set V;

3. H3 holds. Every time an assertional formulae γ^c, such that its signed assertional formulae is of type c, is expanded, then for all objects in $Terms_{Used}$ that have not already been used with γ^c, if $\gamma_1^c \in S$ then $\gamma_2^c \in S$. Hence, H3 holds;

4. H4 holds. Every time an assertion γ^d, such that its signed assertion is of type d, is expanded, then the following hold:

   (a) if $\mathtt{Cl}(o) \notin \Omega$ then x is added to $Terms_{Used}$ and the instance is added to the upper sequent;

   (b) otherwise, since the path does not end into an axiom, neither $\gamma = \bot(o)$ nor $\delta = \top(o)$, the particular case wrt an individual is added to the upper sequent.

   Hence, H4 is satisfied and the claim holds. □

By Lemma 15 there is an interpretation $\mathcal{I}$ which satisfies S. Note that by definition of sequent, $\Sigma \subseteq \Gamma$, hence $\mathcal{I}$ satisfies Σ. By construction of S and from the rules, it follows that $\mathcal{I}$ (Σ, Ω)-satisfies Γ. This implies that $\mathcal{I}$ falsifies $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

Note that H must be infinite if the tree T is infinite. Otherwise, since **Search** starts with a finite sequent, every path would be finite and would end either with an axiom or a finished sequent.

If the sequent $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid then the tree T must be a proof tree since otherwise, the above argument shows that $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is falsifiable. □

B.6 Proofs of Section 5.4

Theorem 8 (Four-valued interpolation theorem) $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ *is a valid sequent iff one of the following cases hold:*

1. *there exists a constructible interpolant* γ^* *of* $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$*;*
2. Γ *is not* (Σ, Ω)*-satisfiable;*
3. $(\bigvee_{\delta \in \Delta}) \equiv \top(a)$*, for some* a*.* ∎

Proof: The *(if)* direction is straightforward.

(Only if) direction. If $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is valid then there is a proof tree T of which it is the conclusion. We use the induction principle applied to the depth n of proof trees.

case n=0: In this case $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is an axiom.

case $\alpha, \Gamma \rightarrow_{(\Sigma,\Omega)} \alpha, \Delta$: $\gamma^* = \alpha$ is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$

$\Gamma \rightarrow_{(\Sigma,\Omega)} \top(o), \Delta$: therefore, case (3) of the theorem holds;

$\bot\,(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$: therefore, case (2) of the theorem holds;

$\Gamma \rightarrow_{(\Sigma,\Omega)} \neg A(a), \Delta$: $\gamma^* = \neg A(a)$ is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$;

$\Gamma \rightarrow_{(\Sigma,\Omega)} \neg P(a,b), \Delta$: $\gamma^* = \neg P(a,b)$ is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

induction step: Suppose that for every proof tree of depth n, the conclusion is valid. We will show that still this is true for proof trees of depth $n+1$. Let T be a proof tree of depth $n+1$ with conclusion $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$. Then T has one of the following forms:

form 1: T is of the form

$$\frac{T'}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

if the conclusion $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is obtained by application of an one-premise rule R, where the conclusion of the proof tree T', the sequent S, is the premise of the rule R and $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is the conclusion of the rule R;

form 2: T is of the form

$$\frac{T' \quad T''}{\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta}$$

if the conclusion $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is obtained by application of a two-premises rule R, where the conclusions of the proof trees T' and T'', respectively the sequents S' and S'', are the premises of the rule R and $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is the conclusion of the rule R.

We proceed by case analysis on the rules of Definition 35.

case rule $(\wedge \rightarrow)$**:** In this case T is of form 1, $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is of form $\gamma \wedge \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ and the sequent S is of form $\gamma, \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$. By induction on tree T', one of the three cases of the theorem holds for $\gamma, \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$. Hence, one of the three cases of the theorem holds for $\gamma \wedge \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$, too.

case rule $(\rightarrow \wedge)$**:** In this case T is of form 2, $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is of form $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, \gamma \wedge \delta$, S' is of form $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, \gamma$ and S'' is of form $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, \delta$. By induction on T' and T'', one of the three cases of the theorem holds. Therefore,

1. if the antecedent of S' is not (Σ,Ω)-satisfiable, then the antecedent of the conclusion is not (Σ,Ω)-satisfiable;
2. if both consequents of S' and S'' are equivalent to $\top(a)$, for some a, then the consequent of the conclusion is equivalent to $\top(a)$;
3. otherwise, if the consequent of S' is not equivalent to $\top(a)$, then there exist interpolant γ' of S'; if the consequent of S'' is not equivalent to $\top(a)$, then there exist interpolant γ'' of S''. Let γ^* be $\gamma' \wedge \gamma''$. γ^* is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

Hence, one of the three cases holds for $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

case rule $(\vee \rightarrow)$**:** In this case T is of form 2, $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is of form $\gamma \vee \delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$, S' is of form $\gamma, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ and S'' is of form $\delta, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$. By induction on T' and T'', one of the three cases of the theorem holds. Therefore,

1. if both the antecedents of S' and S'' are not (Σ, Ω)-satisfiable, then the antecedent of the conclusion is not (Σ, Ω)-satisfiable;
2. if the consequent of S' is equivalent to $\top(a)$, for some a, then the consequent of the conclusion is equivalent to $\top(a)$;
3. otherwise, if the antecedent of S' is (Σ, Ω)-satisfiable, then there exist interpolant γ' of S'; if the the antecedent of S'' is is (Σ, Ω)-satisfiable, then there exist interpolant γ'' of S''. Let γ^* be $\gamma' \vee \gamma''$. γ^* is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

Hence, one of the three cases holds for $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

case rule $(\rightarrow \vee)$**:** Similar to case rule $(\wedge \rightarrow)$.

case rule $(\bot_{C1} \rightarrow)$**:** Case (2) of the theorem holds.

case rule $(\bot_{C2} \rightarrow)$**:** Case (2) of the theorem holds.

case rule $(\bot_{R1} \rightarrow)$**:** Case (2) of the theorem holds.

case rule $(\bot_{R2} \rightarrow)$**:** Case (2) of the theorem holds.

case rule $(\sqcap \rightarrow)$**:** Similar to case rule $(\wedge \rightarrow)$.

case rule $(\rightarrow \sqcap)$**:** Similar to case rule $(\rightarrow \wedge)$.

case rule $(\sqcup \rightarrow)$**:** Similar to case rule $(\vee \rightarrow)$.

case rule $(\rightarrow \sqcup)$**:** Similar to case rule $(\rightarrow \vee)$.

case rule $(\forall \rightarrow)$**:** In this case T is of form 1, $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is of form $(\forall R.C)(t), R(o, o'), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ and the sequent S is of form $(\forall R.C)(t), R(o, o'), C(o'), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$. By induction on tree T', one of the three cases of the theorem holds for S'. Hence, one of the three cases of the theorem holds for the conclusion, too.

case rule $(\rightarrow \exists)$**:** Similar to the case above.

case rule $(\rightarrow \forall)$**:** In this case T is of form 1, $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is of form $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta, (\forall R.C)(o)$ and S is of form $\Gamma' \rightarrow_{(\Sigma,\Omega)} \Delta, \delta$.

By induction on T' one of the three cases of the theorem holds. Therefore,

1. if the antecedent of S is not (Σ, Ω)-satisfiable, then the antecedent of the conclusion is not (Σ, Ω)-satisfiable;
2. if the consequent of S is equivalent to $\top(a)$, for some a, then the consequent of S is equivalent to $\top(a)$;

otherwise, there exists an interpolant γ' of S. If $\mathtt{Cl}(o) \notin \Omega$ then proceed as follows.

Let γ^* be following transformation of γ':

1. γ' can be rewritten as $(\gamma_1\ op'_1\ \gamma'_1)\ op_1\ (\gamma_2\ op'_2\ \gamma'_2)\ op_2\ \ldots\ (\gamma_n\ op'_n\ \gamma'_n)$ such that $n \geq 1$ and (i) in every assertion occurring in γ_i, x occurs, (ii) γ'_i can not be empty (for $i < n$) and x does not occur in γ'_i and (iii) $op_i, op'_i \in \{\wedge, \vee\}$;
2. for each γ_i, remove occurrences of assertion $R(t, x)$;
3. for each γ_i, replace the $\wedge$ and $\vee$ operators in γ_i with the $\sqcap$ and $\sqcup$ operators, respectively;
4. for each γ_i, replace all assertions $C(x)$ occurring in γ_i with C and let γ_i^c be the obtained concept;
5. for each γ_i^c, replace it with $\forall R.\gamma_i^c$.

γ^* is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

On the other hand, if $\texttt{Cl}(o) \in \Omega$ then $\gamma^* = \gamma'$, is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

Therefore, one of the three cases of the theorem holds.

case rule $(\exists \rightarrow)$: In this case T is of form $1, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ is of form $(\exists R.C)(o), \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$ and S is of form $\gamma, \Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$. By induction on T' one of the three cases of the theorem holds. Therefore,

1. if the antecedent of S is not (Σ, Ω)-satisfiable, then the antecedent of the conclusion is not (Σ, Ω)-satisfiable;
2. if the consequent of S is equivalent to $\top(a)$, for some a, then the consequent of S is equivalent to $\top(a)$;

otherwise, there exists an interpolant γ' of S. If $\texttt{Cl}(o) \notin \Omega$ then proceed as follows. Let γ^* be following transformation of γ':

1. γ' can be rewritten as $(\gamma_1\ op'_1\ \gamma'_1)\ op_1\ (\gamma_2\ op'_2\ \gamma'_2)\ op_2\ \ldots\ (\gamma_n\ op'_n\ \gamma'_n)$ such that $n \geq 1$ and (i) in every assertion occurring in γ_i, x occurs, (ii) γ'_i can not be empty (for $i < n$) and x does not occur in γ'_i and (iii) $op_i, op'_i \in \{\wedge, \vee\}$;
2. for each γ_i, remove occurrences of assertion $R(t, x)$;
3. for each γ_i, replace the $\wedge$ and $\vee$ operators in γ_i with the $\sqcap$ and $\sqcup$ operators, respectively;
4. for each γ_i, replace all assertions $C(x)$ occurring in γ_i with C and let γ_i^c be the obtained concept;
5. for each γ_i^c, replace it with $\exists R.\gamma_i^c$.

γ^* is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

On the other hand, if $\texttt{Cl}(o) \in \Omega$ then $\gamma^* = \gamma'$, is an interpolant of $\Gamma \rightarrow_{(\Sigma,\Omega)} \Delta$.

Therefore, one of the three cases of the theorem holds. □

Bibliography

[Anderson and Belnap, 1975] Alan R. Anderson and Nuel D. Belnap. *Entailment - the logic of relevance and necessity.* Princeton University Press, Princeton, NJ, 1975.

[Artale and Franconi, 1994] 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.

[Artale and Franconi, 1995] 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.

[Baader and Hanschke, 1991] 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.

[Baader and Laux, 1995] 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.

[Belnap, 1977a] Nuel D. Belnap. How a computer should think. In Gilbert Ryle, editor, *Contemporary aspects of philosophy*, pages 30–56. Oriel Press, Stocksfield, GB, 1977.

[Belnap, 1977b] Nuel D. Belnap. A useful four-valued logic. In Gunnar Epstein and J. Michael Dunn, editors, *Modern uses of multiple-valued logic*, pages 8–37. Reidel, Dordrecht, NL, 1977.

[Beneventano *et al.*, 1993] 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.

[Blair, 1990] David C. Blair. *Language and representation in information retrieval.* Elsevier science publishers, 1990.

[Börger, 1988] Egon Börger. *Computability, Complexity, Logic.* North-Holland Studies in Logic and Foundations of Mathematics, North-Holland, 1988.

[Chellas, 1980] Brian Chellas. *Modal Logic, An Introduction.* Cambridge University Press, 1980.

[Cooper, 1971] W. S. Cooper. A definition of relevance for information retrieval. *Information Storage and Retrieval*, 7:19–37, 1971.

[De Giacomo and Lenzerini, 1995] 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.

[Donini *et al.*, 1991a] 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.

[Donini *et al.*, 1991b] 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".

[Donini *et al.*, 1992a] F.M. Donini, M. Lenzerini, D. Nardi, A. Schaerf, and W. Nutt. Adding epistemic operators to concept languages. In *Proceedings of the 3rd Int. Conf. on Principles of Knowledge Representation and Reasoning KR-92*, pages 342–353. 1992.

[Donini *et al.*, 1992b] F.M. Donini, M. Lenzerini, D. Nardi, A. Schaerf, and W. Nutt. Queries, rules and definitions as epistemic sentences in concept languages. In *Proceedings of the ECAI Workshop on Knowledge Representation and Reasoning*, Lecture Notes in Artificial Intelligence. 1992.

[Donini *et al.*, 1992c] 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.

[Donini *et al.*, 1992d] 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.

[Gallier, 1986] Jean H. Gallier. *Logic for Computer Science: Foundations of Automatic Theorem Proving.* Harper & Row Publishers, New York, 1986.

[Gentzen, 1935] G. Gentzen. Untersuchungen über das logische Schliessen. *Mathematische Zeitschrift*, 39:176–210,405–431, 1935.

[Green, 1995a] Rebecca Green. Topical relevance relationships. I. Why topic matching fails. *Journal of the American Society for Information Science*, 9:646–653, 1995.

[Green, 1995b] Rebecca Green. Topical relevance relationships. II. An exploratory study and preliminary typology. *Journal of the American Society for Information Science*, 9:654–662, 1995.

[Hollunder *et al.*, 1990] 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.

[Hollunder, 1994] Bernhard Hollunder. An alternative proof method for possibilistic logic and its application to terminological logics. In *10th Annual Conference on Uncertainity in Artificial Intelligence*, pages –, Seattle, Washington, 1994. R. Lopez de Mantaras and D. Pool.

[Hughes and Cresswell, 1972] G.E. Hughes and M.J. Cresswell. *An Introduction to Modal Logic*. Methuen and Co, 1972.

[Huibers and Bruza, 1994] T.W.C. Huibers and P.D. Bruza. Investigating aboutness axioms using information fields. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and development in Information Retrieval*, pages 112–121. ACM, Springer-Verlag, July 1994.

[Huibers and Denos, 1995] Theodorus Huibers and Nathalie Denos. A qualitative ranking method for logical information retrieval models. Rapport de recherche MRIM RAP95-005, Groupe MRIM – LGI-IMAG, 1995. This research was partially supported by the Esprit Network of Excellence No.6606 (IDOMENEUS Research Exchange Program), and by CNET with contract No.94-1B034.

[Levesque, 1984] Hector J. Levesque. A logic of implicit and explicit belief. In *Proceedings of AAAI-84, 4th Conference of the American Association for Artificial Intelligence*, pages 198–202, Austin, TX, 1984.

[Lukaszewicz, 1990] W. Lukaszewicz. *Non-Monotonic reasoning*, chapter 7: Approaches to Closed World Assumption, pages 281–308. Ellis Horwood series in Artificial Intelligence. Ellis Horwood, New York, 1990.

[Marron, 1977] M. E. Marron. On indexing, retrieval and the meaning of about. *Journal of the American Society for Information Science*, 28:38–43, 1977.

[Meghini *et al.*, 1993] C. Meghini, F. Sebastiani, U. Straccia, and C. Thanos. A model of information retrieval based on a terminological logic. In *Proceedings of SIGIR-93, 16th ACM Conference on Research and Development in Information Retrieval*, pages 298–307, Pittsburgh, PA, July 1993.

[Nebel, 1990] Bernhard Nebel. *Reasoning and revision in hybrid representation systems.* Springer, Heidelberg, FRG, 1990.

[Nie and Chiaramella, 1990] Janyun Nie and Yves Chiaramella. A retrieval model based on an extended modal logic and its application to the rime experimental approach. In *ACM SIGIR 90 Bruxelles*, pages 25–43, 1990.

[Patel-Schneider *et al.*, 1991] 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.

[Patel-Schneider, 1986] 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.

[Patel-Schneider, 1987a] Peter Patel-Schneider. *Decidable First-Order Logic for Knowledge Representation.* PhD thesis, University of Toronto, May 1987.

[Patel-Schneider, 1987b] Peter F. Patel-Schneider. A hybrid, decidable, logic-based knowledge representation system. *Computational Intelligence*, 3:64–77, 1987.

[Patel-Schneider, 1987c] P.F. Patel-Schneider. *Decidable, Logic-Based Knowledge Representation.* PhD thesis, University of Toronto, May 1987. Also Tech. Rep. 201/87 of the Department of Computer Science, University of Toronto.

[Patel-Schneider, 1988] Peter F. Patel-Schneider. Adding number restrictions to a four-valued terminological logic. In *Proc. of the 7th Nat. Conf. on Artificial Intelligence (AAAI-88)*, pages 485–490, 1988.

[Patel-Schneider, 1989] Peter F. Patel-Schneider. A four-valued semantics for terminological logics. *Artificial Intelligence*, 38:319–351, 1989.

[Popkorn, 1994] S. Popkorn. *First steps in modal logic.* Cambridge univerity press, 1994.

[Reiter, 1978] 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.

[Reiter, 1984] 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.

[Reiter, 1987] R. Reiter. On closed-world data bases. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 55–76. Plenum Press, 1987.

[Reiter, 1990a] R. Reiter. On asking what a database knows. In J.W. Lloyd, editor, *Proceedings of the Symposium on Computational Logic*, Commission of the European Communities, DG XIII, Esprit Basic Research Series, pages 96–113. Springer Verlag, Nov 1990.

[Reiter, 1990b] 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.

[Saracevic, 1976] T. Saracevic. Relevance: A review of the literature and a framework for thinking on the notion in information science. *Advances in Librarianship*, 6:79–138, 1976.

[Schmidt-Schauß and Smolka, 1991] Manfred Schmidt-Schauß and Gert Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48:1–26, 1991.

[Schmiedel, 1990] 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.

[Schmolze, 1989] 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.

[Sebastiani, 1994] 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.

[Straccia, 1993] 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.

[Straccia, 1995] Umberto Straccia. Four-valued terminological logics for information retrieval. Technical report, Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Pisa, Italy, 1995. FERMI Report 7/95.

[Thistlewaite *et al.*, 1988] Paul B. Thistlewaite, Michael A. McRobbie, and Robert K. Meyer. *Automated Theorem-Proving in Non-Classic Logics*. Pitman, London, GB, 1988.

[van Rijsbergen, 1986a] Keith 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.

[van Rijsbergen, 1986b] Keith van Rijsbergen. A non-classical logic for information retrieval. *The Computer Journal*, 29:481–485, 1986.

[Yao, 1995] Y. Y. Yao. Measuring retrieval effectiveness based on user preference of documents. *Journal of the American Society for Information Science (JASIS)*, 46(2), March 1995.

[Yen, 1991] 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.