ABox Abduction in Description Logic

MSc Thesis (*Afstudeerscriptie*)

written by

Szymon Klarman
(born September 18th, 1981 in Łódź, Poland)

under the supervision of **Dr Ulle Endriss**, and submitted to the Board of Examiners in partial fulfillment of the requirements for the degree of

MSc in Logic

at the *Universiteit van Amsterdam*.

Date of the public defense:	**Members of the Thesis Committee:**
June 27, 2008	Prof. Dr Peter van Emde Boas
	Dr Ulle Endriss
	Dr Stefan Schlobach
	Dr Radboud Winkels

Institute for Logic, Language and Computation

Abstract

In this thesis we elaborate on logic-based automated reasoning techniques for abduction, driven by the principle of goal-oriented reasoning. In the first part we develop two variants of a computational framework for abduction in propositional logic, based on regular connection tableaux and resolution with set-of-support. The procedures are proven to be sound and complete calculi for finding consistent, minimal and relevant solutions to abductive problems. In the second part we adapt the framework to the Description Logic $\mathcal{ALC}$. We obtain a procedure for solving ABox abduction problems (i.e. abductive problems whose main part of the input and every solution are specified by a set of ABox assertions), for which we prove the results of (plain) soundness and (minimality) completeness.

Contents

Chapter 1

Introduction

In recent decades abduction has gained considerable attention in such fields as logic, artificial intelligence and philosophy of science. It has been widely recognized that the style of reasoning, usually illustrated as an inference from puzzling observations to explanatory hypotheses, is in fact inherent in a vast majority of problem-solving and knowledge acquisition tasks. The scope of applications is immense and varies from scientific discovery, through medical and engineering diagnosis, design problems, planning, language and multimedia interpretation, to example generation in tutoring systems, and many others.

In the face of such a widespread and diverse application interest much research has been devoted to elaborating a better understanding of the theoretical grounds underlying abductive reasoning [Aliseda-Llera, 1997, Schurz, 2002, Flach and Kakas, 2000], and on the other hand — to developing computational frameworks for abduction (especially in the context of logic programming, cf. [Kakas et al., 1992, Endriss et al., 2004]) or abduction-based algorithms addressing specific tasks. From the logical perspective, particularly important work has been presented in [Mayer and Pirri, 1993, Mayer and Pirri, 1995], [Aliseda-Llera, 1997], where foundations for tableaux-based proof systems for abduction in propositional, modal and first-order logic have been laid. Prior to these, quite different approaches to abduction in logic, built on linear resolution, have been investigated and employed in several applications [Paul, 1993].

The general goal of this work is to gather some of the results and experiences obtained in the research on abduction so far, and adapt them and explore their value in yet another field: Description Logic (DL). Undoubtedly, DL has become nowadays a leading paradigm of logic-based knowledge representation — the process intensively fueled by the efforts on the development of the Semantic Web, underpinned by the DL-based Web Ontology Language (OWL)[1]. Due to the growing popularity of the formalism, the demand raises for efficient tools providing diverse reasoning services for DL knowledge bases. Whereas highly optimized deductive reasoning algorithms for expressive DLs are already available

[1] `http://www.w3.org/TR/owl-features/`.

(cf. [Horrocks, 1998, Haarslev and Möller, 1999, Kazakov and Motik, 2008]), the advances on non-monotonic types of inference — in particular on abduction — are still very limited, though need for them is obvious. In a programmatic paper [Elsenbroich et al., 2006], the authors advocate for setting off the research on abduction in the context of DL ontologies, supporting the argumentation with several application scenarios. To generalize that overview, it is enough to observe that any research community that reveals an interest in employing DL/Semantic Web technologies to its own domain, while relying on abduction as a part of its methodological toolkit (e.g. e-Science, medical informatics, law and AI), can potentially benefit from the results of a study on abduction in DL.

The contribution of this thesis is twofold. First, we elaborate on logic-based approaches to automated abductive reasoning known from the literature, and devise a general computational framework for finding complete sets of solutions to abductive problems, where the forms of both input and output knowledge are constrained with certain basic logical and methodological requirements.

The framework accounts for two computational variants: one based on semantic tableaux, the other on the resolution technique. The novelty within this part of the research includes several refinements to the existing procedures, stemming predominantly from the adoption of the principle of a *goal-oriented reasoning*. Our strong belief is that this principle is indispensable to processes of abductive reasoning, given their fundamental intractability, and so it should drive the procedure at all stages, involving the actual search for explanations, but also the translation of relevant parts of the knowledge base into a computationally manageable form, and the process of checking consistency of the generated solutions against the background knowledge. The approach, resting on the notion of *connection* (a pair of complementary literals), allows for identifying and a convenient retrieval of only those formulas from the knowledge base that have good chances of contributing to the solution. In this respect, the advocated perspective is deeply concerned with efficiency, or one should rather say — *control* issues, but only on the most general, architectural level, so that universality and flexibility of the approach is respected to the maximum possible extent. Consequently, we shall not consider any advanced and sophisticated techniques being constantly introduced in the field of automated reasoning, leaving open the possibility of improving the framework in the potential implementations, by employing algorithms with most optimized performance ratings.

The second, and central contribution of the work is the adaptation of the developed framework to Description Logic. More specifically, we focus on ABox abduction, which embraces reasoning problems whose input and output can be expressed as a set of ABox assertions, i.e. pieces of *factual knowledge* contained in a DL knowledge base. This part of the investigation is especially exciting, as abduction in the context of DL knowledge bases remains so far a practically uncharted territory. Even so, it might seem that DL, being a subset of first-order logic and a notational variant of multimodal logic K, should straightforwardly accommodate the results obtained for those formalisms, without introducing extra problems. Unfortunately, this would be a too naive expectation. In fact,

the interplay between peculiar expressiveness of DL, witnessed already in its basic language $\mathcal{ALC}$, the syntactic constraints on the admissible form of abductive explanations characterizing ABox abduction, and finally, the nature of the control strategies employed in the abductive framework, gives rise to many unexpected difficulties. Hence, the correspondence between DL and other logics — though certainly the cornerstone of our methodology — has its limits and has to be used with caution. Another source of problems, usually not of a main importance in FOL or modal logic, lies in the necessity of using special techniques (so-called *blocking rules*) in order to guarantee termination the reasoning over certain kind of DL knowledge bases. Again, a naive incorporation of the standard solutions into the abductive framework easily fails.

Altogether, in the course of exploring the subject we shall encounter many open problems, requiring a somewhat inventive, or at least considerably revised approach. In all these cases the proposed solutions are implicitly directed by an ultimate objective of providing firm logical and computational foundations for implementing *abductive DL reasoners*, similar to ones offering deductive reasoning services, such as: Pellet, KAON2 or FaCT[2], and possibly built on top of them, for a convenient reuse of existing and well-developed technologies. Naturally, with this work we only attempt to make the first step towards that direction, and inevitably many issues will have to be left open.

The thesis is organized in the following parts. In the next chapter we shortly elaborate on the notion of abduction, and extract its desired meaning, which is used in the remainder of the report. Further, in Chapter 3, we move to the issues concerning methods of automated abductive reasoning, where we discuss tableaux- and resolution-based approaches to computing abductive explanations, and derive their logical properties. In Chapter 4 we provide an introduction to Description Logic and the problem of abduction in its context, thus preparing sufficient grounds for the main part of the thesis, in which we develop the computational framework for ABox abduction in DL. This work is presented in Chapter 5, where focusing on the basic language $\mathcal{ALC}$ we introduce the appropriate syntactic transformation of DL knowledge bases, define the $\vdash_{\text{ABox}}$ inference for solving ABox abduction problems and prove its (restricted) soundness and completeness. In the last chapter we summarize the work with conclusions and propose directions for future work.

[2]See `http://www.cs.man.ac.uk/~sattler/reasoners.html` for an overview.

Chapter 2

Abduction

Abduction is a form of *ampliative* reasoning, i.e. reasoning that allows for acquiring new pieces of knowledge. Typically, it is associated with the following inference scheme:

$$\frac{\begin{array}{c}\alpha \rightarrow \varphi \\ \varphi\end{array}}{\alpha}$$

where $\alpha \rightarrow \varphi$ belongs to background knowledge, φ is a fact that is to be explained, and α is the conclusion: an explanation to φ. As should be already clear from the above diagram, the information content of the conclusions of abductive inferences, unlike in deduction, is not included in the content of their premises. Naturally, this powerful feature comes at a price, namely of inferring a false conclusion. Because of that, abduction is also often classified as a form of *hypothetical* or *fallible* reasoning: we are conjecturing that α might be the case, but cannot be sure of that.

The presence of abduction in philosophy and logic traces back to works of Aristotle, but its fully acknowledged position as one of the three basic modes of inference, alongside deduction and induction, and its formal characterization, have not been established until Charles S. Pierce.[1] Since recent literature abounds with philosophical discussions on abduction (cf. [Aliseda-Llera, 1997, Schurz, 2002, Thagard, 1988]) we will skip the broader background here, and instead focus on defining a suitable logical interpretation of the basic setting underlying abductive reasoning.

[1] A comprehensive overview of the origins and evolution of the notion can be found for instance in [Aliseda-Llera, 1997].

2.1 Abductive problems and solutions

Considerations on abduction and the standard nomenclature used in its context are often strongly influenced by the traditional perspective imposed by philosophy of science, where abduction is typically seen as the main tool employed in the process of scientific discovery (see e.g. [Paavola, 2004]). Hence, there is an apparent tendency to think of abductive reasoning as the reasoning from *observations* to *explanations* or from *effects* to *causes*. This heritage, though justified by historical circumstances, becomes quite a burden once the variety of settings in which abduction occurs is acknowledged.[2] As one example, consider the following MacGyver's problem:

> *I have a piece of wire, a watch, a battery and a bottle of petrol. How can I construct a time bomb?*

It would be hard to argue there is some genuinely explanatory or causal reasoning involved in the process of solving the problem. Still, it is clearly abductive (or rather a mixture of abductive and deductive) style of reasoning that allows to guess a promising composition of elements, which makes up the object with all the desired properties. Similarly, abduction is used by human reasoners for scheduling optimal plans, generating examples, or interpreting abbreviated sms messages — problems quite remote from the classically invoked frameworks of scientific explanation or medical diagnosis.

On the formal side, the impact of the traditional view on abduction can be observed in the role frequently assigned to material implication in models of abductive inference [Paul, 1993]. It is usually assumed that what there is to be explained has to occur as the consequent of an implication, whereas its antecedent is a plausible explanation, as if the implication connective was representing some form of explanatory or causal relationship. This of course might be the case if one takes it as her knowledge modeling policy, but it certainly does not follow from the logical formalism itself. Notice, that the inference scheme presented in the beginning of this chapter could be equally well rewritten in any of the two forms below, without the loss of logical soundness, in the sense of maintaining the isomorphism of (classical) entailment relationships between the premises and the conclusion.

$$\frac{\neg\alpha \vee \varphi \quad \varphi}{\alpha} \qquad \frac{\neg\varphi \rightarrow \neg\alpha \quad \varphi}{\alpha}$$

To avoid any confusion as to the question of what abduction really is about, we emphasize that in this report we want to abstract from the dominant traditional approach, both interpretative and representational, while dealing with

[2] The variability of the abductive context has been recently highlighted in [Gabbay and Woods, 2006].

abduction and try to stay as close as possible to its broadest logical meaning, captured in the two definitions below.

Definition 1 (Abductive problem)**.** An *abductive problem* is a pair $\langle \mathcal{K}, \varphi \rangle$, where $\mathcal{K}$ is a set of logical formulas (the knowledge base), φ is a logical formula (the query), and $\mathcal{K} \nvDash \varphi$.

Definition 2 (Plain abduction)**.** A logical formula α is a *plain solution* to an abductive problem $\langle \mathcal{K}, \varphi \rangle$ *iff* $\mathcal{K} \cup \{\alpha\} \vDash \varphi$.

Notice, that in the explanationist context we would call φ the *explanandum* and α an *explanans* of an abductive problem. Instead we will use the notions of *the query* of a problem and *a solution* to a problem, or in some cases, a solution to the query, whenever the identity of the knowledge base, relative to which the solution is formulated, is obvious.

Clearly, the condition $\mathcal{K} \nvDash \varphi$, included in the first definition, is the only constitutive feature of abductive problems, whereas the entailment property involved in the second definition is the only determinant of plain solutions to those problems. We do not assume anything about the knowledge base, apart from the fact that the formulas contained in it are well-formed, with respect to the syntax of some logical language, and are implicitly connected by conjunction.

The essence of abductive reasoning in this broadened perspective comes down to finding a formula α that added to the knowledge base $\mathcal{K}$ enforces $\mathcal{K} \cup \{\alpha\}$ (either by eliminating all undesired models of $\mathcal{K}$ or augmenting them with necessary additions) to entail φ, which otherwise would not be the case. Our claim is that plain abduction does comprise the logical core of abductive reasoning, and other, more restrictive interpretations follow either from accepting additional, epistemologically justified constraints (some of which will be introduced below) or from application-driven adjustments. If one nevertheless keeps requesting a more informal reading of the abductive setting scoped as presented above, we suggest to resort to the following, epistemologically and ontologically neutral (but hence quite vacuous) phrase: *α solves $\langle \mathcal{K}, \varphi \rangle$ iff knowing α and $\mathcal{K}$ is sufficient to know that φ is true, whereas knowing $\mathcal{K}$ alone is not.*

2.2 Formal constraints

In principle not all plain solutions are of interest to the reasoner. In fact, infinitely many are not even considered as they are inconsistent, irrelevant or include more information than necessary. Given the aim of devising a generic computational procedure for finding all interesting solutions, it is therefore advisable to strengthen the selection criteria for abductive solutions with all feasible constraints that allow for safe delimitation of the search space. Ideally, safeness should be warranted here by the reference to only the most rudimental logical and epistemological principles. The choice proposed below, commonly occurring in the literature (cf. [Aliseda-Llera, 1997, Paul, 1993]), should be to our opinion the least controversial, as it embraces arguably the most intuitive and universal criteria used in all applications of abductive reasoning.

Definition 3 (Consistency, relevance, minimality)**.** Let α be a plain solution to an abductive problem $\langle \mathcal{K}, \varphi \rangle$. We call α:

1. *consistent* iff $\mathcal{K} \cup \{\alpha\} \nvDash \bot$.
2. *relevant* iff $\alpha \nvDash \varphi$.
3. *minimal* iff for every consistent and relevant solution β if $\alpha \vDash \beta$ then $\beta \vDash \alpha$.

The consistency requirement leads to discarding all those solutions that are inconsistent with the knowledge base (or even just by themselves). For instance, let $\mathcal{K} = \{p \to q, \neg p\}$ and $\varphi = q$. Clearly, p is a plain solution to q, but accepting p would contradict a proposition that is already assumed to be true, hence it cannot be considered as a promising hypothesis, unless a revision process is to be involved. The obvious premise behind the criterion is that although abduction is an intrinsically fallible form of reasoning, it is still irrational to accept conclusions that are necessarily false. Another consequence of the criterion, is that abductive problems based on inconsistent knowledge do not have consistent solutions.

Further, it seems reasonable to consider only relevant solutions, i.e. such formulas, which do not entail the query on their own, but need to engage some part of the already accepted body of knowledge in order to do so. Essentially, the requirement prevents from accepting ad hoc solutions, which actually avoid the problem rather than solve it. An extreme case of irrelevance occurs when the query itself is considered as a solution to an abductive problem. Such a solution is usually denoted as a *trivial* one. Naturally, syntactic equivalence to the query is not a distinctive enough feature to identify irrelevant solutions, and other forms, depending on the language in which solutions are to be expressed, are also possible. Some characteristic examples are presented in Table 2.1.

Target language	**Query**	**Irrelevant solution**
Propositional logic	p	$(q \to p) \wedge q$
First-order logic	$P(a)$	$\forall_x P(x)$
Modal logic with converse modalities $(\cdot^-)$	p	$\langle r^- \rangle [r] p$

Table 2.1: Sample irrelevant solutions expressed in different languages.

The notion of relevance adopted here (following [Aliseda-Llera, 1997] and [Elsenbroich et al., 2006]) could in fact be renamed to *weak relevance*. Notice, that under this constraint partially irrelevant solutions are still allowed. For instance for $\mathcal{K} = \{q \to r\}$ it is possible to solve $p \wedge r$ with $p \wedge q$, though part of the solution is clearly trivial. A stronger requirement could be defined as follows:

Definition 4 (Strong relevance)**.** Let α be a plain solution to $\langle \mathcal{K}, \varphi \rangle$. We call α *strongly relevant* iff for every γ, δ such that $\varphi \vDash \gamma$ and $\varphi \vDash \delta$ if $\{\alpha\} \cup \{\gamma\} \vDash \delta$ then $\gamma \vDash \delta$

Given appropriate restrictions on the syntactic form of α, φ, γ and δ (see below) such a formulation enforces that no information contained in the query can follow directly from a solution, without the contribution of the knowledge base. In certain situations strong relevance can be too restrictive, as partial solutions can sometimes bear some significant value. On the other hand, as will be shown later, there are some types of weakly relevant (but not strongly relevant) formulas, which are not in practice very interesting. Hence, defining a well-grained notion of relevance remains an open issue.

The minimality requirement deserves a little more attention, as it is often a subject of debates in the literature, and in fact, several different minimality criteria have been proposed. All of them, in their own manner, attempt to provide a formal expression of the idea of *preference for simplicity*, lying at the heart of Occam's razor. A well-known pitfall is that Occam's razor is one of the most open-ended notions in philosophy, therefore, there are numerous, not rarely incompatible intuitions associated with it. Let us list here and shortly discuss some of the possible interpretations of simplicity that have been introduced in the context of abduction. For clarity, we will slightly systematize the taxonomy, reserving the notion of minimality for designating only the property described in Definition 3, and using the term simplicity for more general contexts.

Definition 5 (Variants of simplicity)**.** Let α be a plain solution to an abductive problem $\langle \mathcal{K}, \varphi \rangle$. We call α:

1. $\mathcal{K}$*-minimal* iff for every consistent and relevant solution β if $\mathcal{K} \cup \{\alpha\} \vDash \beta$ then $\mathcal{K} \cup \{\beta\} \vDash \alpha$.

2. *basic* iff there is no consistent and relevant solution to $\langle \mathcal{K}, \alpha \rangle$.

3. *weakest* iff for every consistent and relevant solution β it holds that $\beta \vDash \alpha$.

Observe, that every criterion reflects certain assumption as to what kind and what amount of information should be included in an epistemically valuable solution. $\mathcal{K}$-minimal and basic solutions are discussed in [Paul, 1993] and referred to as least and most presumptive explanations, respectively. The former criterion requires solutions to express only most general hypotheses that can be generated on the grounds of the knowledge base, whereas the latter, conversely, demands the most specific presumptions. Basicness is also implicitly assumed in [Elsenbroich, 2005] and described as the requirement for explanations of "greatest depth". The minimality [Mayer and Pirri, 1993, Mayer and Pirri, 1995] chooses deductively weakest solutions in the analytical sense of Quine's *prime implicants* [Quine, 1959], i.e. regardless of the assumed relationships between propositions. Finally, the criterion of the weakest solution appears in [Aliseda-Llera, 1997], and in fact could be rephrased as the requirement for (semantic) uniqueness of minimal solution.

Having approved these interpretations, let us examine a simple abductive problem $\langle \mathcal{K}, q \rangle$, assuming the following knowledge base:

$$\begin{aligned} \mathcal{K} = \{ \quad & p \to r, \quad r \to q, \\ & s \wedge \neg u \to t, \quad t \to q, \\ & p \wedge w \to v, \quad v \to q \quad \} \end{aligned}$$

Table 2.2 presents quite surprising outcomes of applying the simplicity criteria to the set of consistent and relevant solutions to the problem. Three out of four criteria, all oriented towards capturing deductive weakness, select the same odd solution $\mathcal{K} \to q$. The reason is that for any solution α it holds by definition that $\mathcal{K} \cup \{\alpha\} \models \varphi$, which in turn entails $\alpha \models \mathcal{K} \to \varphi$. Hence, $\mathcal{K} \to \varphi$ is the weakest of all possible solutions in the sense of all three criteria.

Simplicity criterion	**Selected solutions to** $\langle \mathcal{K}, q \rangle$
minimality	$\mathcal{K} \to q$
weakest solution	$\mathcal{K} \to q$
$\mathcal{K}$-minimality	$\mathcal{K} \to q$
basicness	$p \wedge s \wedge \neg u \wedge w$

Table 2.2: Simple solutions in the full propositional language.

$\mathcal{K} \to \varphi$, although not irrelevant, is nevertheless totally uninteresting. On the other hand, solution $p \wedge s \wedge \neg u \wedge w$, returned by the basicness criterion, is also quite counterintuitive. Clearly, there is some parameter missing from the representation of the abductive setting. This parameter is the specification of the syntactic form for admissible solutions. The most commonly accepted proposal (cf. [Aliseda-Llera, 1997, Mayer and Pirri, 1993, Mayer and Pirri, 1995, Paul, 1993]), which we adopt in our framework as well, is to restrict the language for expressing solutions only to conjunctions of positive and negative literals. Typically, the same constraint is extended also to the syntactic form of queries.

Simplicity criterion	**Selected solutions to** $\langle \mathcal{K}, q \rangle$
minimality	$p, \quad r, \quad s \wedge \neg u, \quad t, \quad v$
weakest solution	none
$\mathcal{K}$-minimality	$r, \quad t, \quad v$
basicness	$p \wedge s \wedge \neg u \wedge w$

Table 2.3: Simple solutions in the conjunctive language.

Given the additional restriction the resulting sets of selected solutions differ considerably from the previous (see Table 2.3). Notice, that this time there is no weakest solution, as there are several incomparable, minimal ones. Still, the problem of a single basic solution remains, and this effect is difficult to

circumvent. In fact, the case of basicness shows that intuitions behind some seemingly natural criteria are sometimes too hard to cast in a logical formalism. What is originally intended with the basicness criterion is to find a way to extract all the formulas that open the implication chains leading to the query, and list them as a menu of distinct, most specific hypotheses. What is captured formally, however, is the requirement for a deductively strongest solution with respect to the knowledge base, which in the presented example is simply the conjunction of all these formulas. A possible remedy is to apply the basicness criterion on top of minimality, which finally brings results closer to the expectations (see Table 2.4).

Simplicity criterion	**Selected solutions to** $\langle \mathcal{K}, q \rangle$
basicness_{min}	$p, \quad s \wedge \neg u$

Table 2.4: Basic (on top of minimal) solutions in the conjunctive language.

In the light of this sketchy evaluation it should be finally possible to justify our strong preference for the minimality criterion (as opposed to the $\mathcal{K}$-minimality and basicness_{min}) as a proper articulation of Occam's razor in the generic framework for abduction. The straightforward argument is that the minimality criterion is simply the least restrictive criterion of all. The only epistemic bias it exhibits is the preference for non-redundancy of information. More precisely, the minimality criterion discards only those solutions that contain excessive information, where excessiveness is detected regardless of any background knowledge about the world, merely by analytical comparison to other possible solutions. Whenever it is possible to solve an abductive problem by accepting p, it is irrational to assume instead that p and r, since this does not bring an extra epistemic gain. This exactly the kind of situations that are tackled by the minimality criterion.

Informativeness — one of the core epistemic values driving all ampliative forms of reasoning — as considered on this level is evidently different from the informativeness understood as specificity of hypotheses given background knowledge. The latter perspective is also highly relevant for abductive processes, but follows from pragmatical, rather than analytical premises. As remarked in [Paul, 1993], in some contexts there will be a demand for specific solutions only, in other — for general. Supposedly, in yet other, one will require "medium-specific" solutions, a quality that is arbitrarily neglected among considered criteria. Most likely in average application scenarios, the requirement for specificity of information is traded off against the risk of error or against the amount of computational resources available for finding solutions. Even though very interesting and important, these are all aspects concerning a different level of analysis, and thus will not been addressed in the scope of this work.

All through the rest of the report, unless explicitly specified otherwise, we will be concerned only with abductive solutions restricted to conjunctive forms, that are consistent, relevant and minimal in the sense explicated in Definition

3. Likewise, we will consider only abductive problems, in which the query is given in the conjunctive form.

Chapter 3

Goal-oriented Abductive Reasoning

The central idea underlying the approach to solving abductive problems we argue for stems from a simple observation on certain heuristics applied by all rational human reasoners dealing with similar problems. Let us illustrate it with a small example.

$$\begin{aligned}\mathcal{K} = \{\quad & \mathsf{hungry\_Koala} \rightarrow \mathsf{anxious\_Koala},\\ & \mathsf{seismic\_movement} \rightarrow \mathsf{earthquake},\\ & \mathsf{no\_eucalyptus} \rightarrow \mathsf{hungry\_Koala},\\ & \mathsf{earthquake} \wedge \mathsf{under\_sea} \rightarrow \mathsf{tsunami},\\ & \mathsf{no\_rain} \wedge \mathsf{no\_sun} \rightarrow \mathsf{no\_eucalyptus},\\ & \mathsf{earthquake} \rightarrow \mathsf{everything\_shaking} \quad \}\end{aligned}$$

Let us ask now: *given* $\mathcal{K}$, *why could it be the case that* anxious_Koala? A quick glance into the knowledge base should immediately lead to the first presumption: *because* hungry_Koala. The next guess would most likely be: *because* no_eucalyptus; and finally: *because* no_rain *and* no_sun. Interestingly, most people should answer the question without any trouble in the same way, not attempting to hypothesize anything about earthquakes or seismic movements, even though the knowledge base is somewhat mixed up. Apparently, whenever possible, we tend to use backward chaining in the reasoning, which drives the process from the initial goal, along implication chains, to the possible solutions, leaving out on the way everything that does not fit into the chain. The strategy is very powerful and allows to reduce the search space dramatically.

Soundness of goal-oriented abduction, as presented above, hinges on a basic property of the standard propositional semantics. Observe, that every formula α, such that $\alpha \rightarrow \varphi \in \mathcal{K}$, is a plain solution to $\langle \mathcal{K}, \varphi \rangle$. By the same token, every β, such that $\beta \rightarrow \alpha \in \mathcal{K}$, is a plain solution to $\langle \mathcal{K}, \alpha \rangle$, and by the transitivity of material implication, also to the former problem. Basically, by bare use of backward chaining one can effectively identify the set of all plain solutions to

an abductive problem, located along implication chains leading to the query. This form of computation is already quite well understood and explored in the context of abductive logic programming, where knowledge bases are represented by means of Horn clauses [Kakas et al., 1992]. As we claimed, however, we do not want to commit ourselves to a particular way of representing knowledge, and rather have a search procedure that is flexible enough to solve abductive problems based on arbitrarily expressed background theories. For that purpose, a more fundamental insight into the mechanism underlying goal-oriented reasoning is required.

The key concept allowing to capture the essence of goal-oriented reasoning is that of *a connection*: an occurrence of complementary literals in two different formulas in Negation Normal Form. Let us rewrite the part of $\mathcal{K}$ relevant to the Koala problem into NNF and represent it in the form of a matrix whose columns are implicitly connected by conjunction and rows by disjunction.

$$\left[\begin{array}{cccc} \left[\begin{array}{c}\mathsf{anxious\_Koala}\\ \neg\mathsf{hungry\_Koala}\end{array}\right] & \left[\begin{array}{c}\mathsf{hungry\_Koala}\\ \neg\mathsf{no\_eucalyptus}\end{array}\right] & \left[\begin{array}{c}\mathsf{no\_eucalyptus}\\ \neg\mathsf{no\_rain}\\ \neg\mathsf{no\_sun}\end{array}\right] & ?_{\mathbf{abd}} \end{array}\right]$$

Table 3.1: Matrix representation of the Koala problem.

Every horizontal path through the entries of the matrix represents a possible model of the knowledge. Paths containing complementary literals are clearly unsatisfiable. The goal of abduction could be interpreted as finding all such combinations of literals that added to the matrix would close (make unsatisfiable) all but those paths that contain the query. Observe, that the choice of the consecutive formulas used for solving the Koala problem was tightly guided by the path of connections in the matrix: from ¬hungry_Koala to hungry_Koala and from ¬no_eucalyptus to no_eucalyptus. The reasoning stopped, when no more connections could be found. This strategy finds a firm formal justification. Adding a connected formula automatically closes at least one path and introduces a new selection of literals, whose complements (at least potentially) can be used for consistently closing the matrix. On the contrary, extending the matrix with a not connected formula does not contribute to the intended closure, as it does not close any of the existing paths. An attempt of closing the matrix by means of the literals from such an irrelevant formula leads inevitably to non-minimality or inconsistency of the solution.

The research on the connection-based theorem proving was initiated in [Bibel, 1981] and [Andrews, 1981], and has resulted in the following theorem:

Theorem 1 (Connection-based satisfiability [Bibel, 1981])**.** *A propositional formula in NNF is unsatisfiable iff there is a spanning set of connections for its matrix.*

In other words, a formula is unsatisfiable if and only if every horizontal path through the matrix representation of its NNF form contains a connection.

The abductive procedures discussed in this thesis are built upon the proving techniques whose mechanisms are explicitly directed by the search for connections, and which thus provide decision procedures for satisfiability interpreted in the sense defined in the above theorem. A quite different approach to goal-oriented abduction has also been investigated in [Elsenbroich, 2005], but instead, founded on an original, goal-oriented deductive proof system elaborated in [Gabbay, 1998]. Whereas this procedure has not been as far commonly acknowledged in the field of automated reasoning, and as such remains a non-standard system, our proposals are based on refinements of two techniques that are most popular and most frequently used in practical applications: *semantic tableaux* and *resolution*.

In the following sections we present the two methods for abductive problem solving and prove their soundness and completeness. We focus only on the propositional case, postponing all issues concerning first-order and modal logic until the next chapter, where we start adapting the framework to Description Logic. We assume acquaintance with the basics of the underlying semantic tableaux and resolution techniques; in particular we will not repeat the soundness and completeness results for both procedures, which can be found for instance in [Hähnle, 2001] and [Bachmair and Ganzinger, 2001].

3.1 Preliminaries

For the sake of the following three sections we assume that the knowledge base of an abductive problem is given in the propositional clause form, defined in the standard way as follows.

Definition 6 (Clause propositional knowledge base)**.** $\mathcal{K}$ is a *clause propositional knowledge base* iff for every $\psi \in \mathcal{K}$, $\psi = \psi_1 \vee \ldots \vee \psi_n$, where every φ_i is a literal, i.e. either an atomic proposition or the negation of an atomic proposition.

Naturally, any propositional knowledge base can be transformed into the clause form by reduction to Conjunctive Normal Form and splitting the conjuncts. Although both discussed procedures operate on clauses only, the requirement for a thorough preprocessing of the knowledge base could be an object of criticism, as it is computationally expensive. For that reason, in Section 3.4, we suggest a more efficient transformation scheme, which can be easily incorporated into the framework.

Conventionally, with every conjunction $\alpha = \alpha_1 \wedge \ldots \wedge \alpha_n$ we associate the set of its conjuncts $Cn_\alpha = \{\alpha_1, \ldots, \alpha_n\}$, and with every clause $\psi = \psi_1 \vee \ldots \vee \psi_n$ the set of its disjuncts $Cl_\psi = \{\psi_1, \ldots, \psi_n\}$. Let us recall now several basic definitions and properties that will be useful in the consequent discussions.

Proposition 1. *For any two propositional conjunctive formulas α and β, such that α and β are satisfiable (none of them contains two complementary literals), it holds:*

1. $\alpha \vDash \beta$ *iff* $Cn_\beta \subseteq Cn_\alpha$

2. $\alpha \vDash \beta$ *iff* $Cl_{\neg\beta} \subseteq Cl_{\neg\alpha}$

The validity of the equivalences is obvious given we deal only with conjunctions and disjunctions of literals. This simple property is very helpful in the abductive setting as it provides a convenient way of verifying relevance and minimality of solutions. Further, we define the notion of a minimally unsatisfiable set.

Definition 7 (Minimal unsatisfiability)**.** A set of clauses is *minimally unsatisfiable (mu)* if it is unsatisfiable and each of its proper subsets is satisfiable. A clause Cl is *relevant* in a set of clauses S if it belongs to a mu subset of S.

Proposition 2. *Every unsatisfiable set of clauses has a finite mu subset.*

Proof. By the compactness theorem every unsatisfiable set of clauses has a finite subset which is unsatisfiable. Let S denote this subset. Since every singleton subset of S is satisfiable, then by induction on the natural numbers we conclude there has to be a finite subset of S which is minimally unsatisfiable. □

Finally, we are ready to derive a simple lemma, which plays a pivotal role in the proofs of soundness and completeness for both abductive procedures.

Lemma 1. *Let* α *be a consistent, relevant and minimal solution to* $\langle \mathcal{K}, \varphi \rangle$*. There exists a finite non-empty set* $S \subseteq \mathcal{K}$ *such that* $S \cup Cn_\alpha \cup \{\neg\varphi\}$ *is mu.*

Proof. Since α is a plain solution to $\langle \mathcal{K}, \varphi \rangle$ it follows that $\Theta = \mathcal{K} \cup Cn_\alpha \cup \{\neg\varphi\}$ is unsatisfiable (Def. 2). Therefore, there exists a finite mu subset of A (Prop. 2). Let us consider some necessary elements of Θ. It is not possible to remove $\neg\varphi$, since by consistency of α we know that $\mathcal{K} \cup Cn_\alpha$ is satisfiable. Also, there is no such formula β that $Cn_\beta \subset Cn_\alpha$ and $\mathcal{K} \cup Cn_\beta \cup \{\neg\varphi\}$ is unsatisfiable, as that would indicate that α is not minimal as was assumed (Def. 3, Prop. 1). Hence, any mu subset of A has to subsume $Cn_\alpha \cup \{\neg\varphi\}$ plus a finite set $S \subseteq \mathcal{K}$. Moreover, S has to contain at least one element or otherwise $Cn_\alpha \cup \{\neg\varphi\}$ would be unsatisfiable, meaning that α is an irrelevant solution, which contradicts the assumption (Def. 3). □

3.2 Tableaux-based approach

A *clause tableau* is a labeled tree whose nodes are clauses or literals. It is started with the root containing a set of clauses, and developed by consecutive applications of *beta expansion rule* to the clauses. Each clause can be expanded only once on a branch. Whenever a branch contains complementary literals, the *closure rule* can be applied (see Table 3.2). A tableau is *saturated* iff no more expansion steps are possible. A tree T is a *tableau refutation proof* of φ from $\mathcal{K}$, denoted as $\mathcal{K} \vdash_{\mathcal{T}} \varphi$ iff the root of T contains $\mathcal{K} \cup \{\neg\varphi\}$, where $\neg\varphi$ has been the first clause to be expanded in T or simply the one used at the bottom of the root (in case it is a unit clause), and all branches of T have been closed. With every tableau T we associate the set of its open branches $\mathbf{\Gamma}_T$, where each

$\Gamma \in \mathbf{\Gamma}_T$ is represented as the set of all and only those literals that occur on the respective branch.

$$\frac{Cl = \{L_1, \ldots, L_n\} \quad \vdots}{L_1 \mid \ldots \mid L_n} \qquad \frac{L \quad \vdots \quad \overline{L}}{\bot}$$

$\beta -$ **rule** **Branch closure**

Table 3.2: Clause tableau rules.

We begin with reviewing a general tableaux-based approach to abduction, as it attracted quite a lot of attention in the literature (cf. [Aliseda-Llera, 1997, Mayer and Pirri, 1993, Mayer and Pirri, 1995]), and remains a direct basis for the refinement proposed in the remainder of the section. The summary, presented below, follows [Mayer and Pirri, 1993].

Essentially, the idea behind the method comes down to enforcing a tableau's closure. By the definition of a tableau proof, a formula φ is entailed by a set of propositional formulas $\mathcal{K}$ if and only if all branches of the saturated tableau tree for $\mathcal{K} \cup \{\neg\varphi\}$ are closed. If this is true then abduction is unnecessary, as the query already follows form the knowledge base. In the opposite case, one can always try to find a formula that added to the remaining open branches will close them. Such a formula will be a plain solution to an abductive problem $\langle \mathcal{K}, \varphi \rangle$.

Let then T be a saturated tableau tree with a non-empty set of open branches. It is easy to notice that any $\Gamma \in \mathbf{\Gamma}_T$ can be closed simply by choosing any literal from it and adding its complement to the branch. Generalizing the procedure, we can define a choice function f over $\mathbf{\Gamma}_T$, which selects exactly one literal from each open branch, whose complement becomes a conjunct of the formula closing the tableau. This insight leads to the following redefinition of abduction:

Definition 8 ($\vdash_{\mathcal{T}}$-abduction)**.** Let $\langle \mathcal{K}, \varphi \rangle$ be an abductive problem and T a saturated tableau tree for $\mathcal{K} \cup \{\neg\varphi\}$. For every choice function f over $\mathbf{\Gamma}_T$, a formula α such that $\vDash \alpha \leftrightarrow \bigwedge_{\Gamma_i \in \mathbf{\Gamma}_T} \neg f(\Gamma_i)$ is a *plain solution* to $\langle \mathcal{K}, \varphi \rangle$. Further, we call α:

1. *consistent* iff it does not close all open branches of the tableau tree for $\mathcal{K}$, i.e. $\mathcal{K} \nvdash_{\mathcal{T}} \neg\alpha$
2. *minimal* iff for every $\alpha_i \in Cn_\alpha$, there is at least one open branch in T such that α_i closes it while no other $\alpha_{j \neq i} \in Cn_\alpha$ does.

The missing relevance requirement can be covered straightforwardly by $\subseteq$-ordering comparison of α and φ. By Definition 3 and Proposition 1 it follows that $Cn_\varphi \subseteq Cn_\alpha$ if and only if α is irrelevant. Hence, we will leave this requirement aside for a moment as less interesting.

It is easy to verify that the above characterization preserves the meaning of the original conditions. Moreover, assuming we perform an exhaustive search through the space of all possible choice functions over $\mathbf{\Gamma}_T$, the method indeed allows for an exact enumeration of all consistent and minimal solutions to an abductive problem. We prove this in the following theorem,[1] relying on soundness and completeness of the tableau method:

Theorem 2 ($\vdash_{\mathcal{T}}$-abduction: soundness and completeness)**.** *A formula α is a plain (1), consistent (2) and minimal (3) solution to the problem $\langle \mathcal{K}, \varphi \rangle$* iff *$\alpha$ is a $\vdash_{\mathcal{T}}$-plain, consistent and minimal solution to $\langle \mathcal{K}, \varphi \rangle$.*

Proof. ($\Rightarrow$) Assume α is a plain, consistent and minimal solution to $\langle \mathcal{K}, \varphi \rangle$. (1) Since α is a plain solution to $\langle \mathcal{K}, \varphi \rangle$, i.e. if $\mathcal{K} \cup \{\alpha\} \vDash \varphi$ it follows that the tableau for $\mathcal{K} \cup \{\alpha\} \cup \{\neg\varphi\}$ is closed. Unless α contains a pair of complimentary literals (self-contradiction), which is prohibited by the consistency requirement, then it can close the tableau only if it contains at least one complementary literal from each open branch. But then there exists a choice function that picks them from branches. Thus if α does not contain any irrelevant literals (in that case it would be non-minimal) then it can be found by search through the space of all possible choice functions. (2) Since α is a consistent solution, i.e. $\mathcal{K} \nvDash \neg\alpha$, surely the tableau for $\mathcal{K} \cup \{\alpha\}$ will not be closed. (3) Assume that the condition for $\vdash_{\mathcal{T}}$ minimality does not hold. Then there is one conjunct of α, say α_i, for which there is no such open branch in the tableau that would be closed exclusively by α_i. Notice, that then α_i is actually not necessary for the tableau to be closed, since a shorter conjunction β, such that $Cn_\beta = Cn_\alpha \setminus \{\alpha_i\}$, suffices. Clearly, $\alpha \vDash \beta$ but $\beta \nvDash \alpha$, which contradicts the assumed minimality of α (Def. 3, Prop. 1).

($\Leftarrow$) Assume α is a $\vdash_{\mathcal{T}}$-plain, consistent and minimal solution to $\langle \mathcal{K}, \varphi \rangle$. (1) Since α has been generated by means of some choice function over $\mathbf{\Gamma}_T$ it clearly follows that the tableau for $\mathcal{K} \cup \{\alpha\} \cup \{\neg\varphi\}$ is closed. This means that $\mathcal{K} \cup \{\alpha\} \vDash \varphi$ and so that α is a plain solution to $\langle \mathcal{K}, \varphi \rangle$. (2) α is consistent with $\mathcal{K}$, because it is guaranteed that the tableau $\mathcal{K} \cup \{\alpha\}$ remains open. (3) In the first part of the proof we have shown that all minimal solutions are indeed found be the procedure. Now we have to show, that only those solutions are actually selected. If α is $\vdash_{\mathcal{T}}$-minimal, then leaving out any of its conjuncts will eventuate in a failure to close at least one open branch. Note, that $\mathbf{\Gamma}_T$ is invariant to the order of expansion steps, hence this consequence holds for any saturated tableau for $\mathcal{K} \cup \{\neg\varphi\}$. Since we consider only solutions in conjunctive form, we see that there is no other formula β entailed by α alone (i.e. $Cn_\alpha \subseteq Cn_\beta$) that would solve the same abductive problem (Def. 3, Prop. 1). Hence α is indeed minimal. □

[1] Proof of the theorem is omitted in both [Mayer and Pirri, 1993] and [Aliseda-Llera, 1997]. An interesting aspect of it, and also of the remaining ones presented in this chapter, is that its "soundness" part, depends essentially on the "completeness". This peculiar effect is due to the characterization of minimality, defined by reference to all possible solutions. Hence, one has to first prove that all "really" minimal solutions are indeed captured by the search procedure, before ensuring that the strategy of verifying minimality used in the particular setting correctly singles them out. Without minimality criterion it is also impossible to prove soundness of plain abduction, as there is infinitely many plain solutions.

To give a feeling of how inefficient the method could be, consider that for the saturated tableau for the Koala problem, opening this chapter, there are 7^{14} possible choice functions over the set of its open branches. Even if there were appropriate refinements introduced, which would allow for deterministically picking only minimal solutions, there would still remain around 10 solutions, out of which only 3 are in fact consistent and relevant. Leaving out the single irrelevant answer, one has to run the consistency checking procedure 6 times in order to obtain the final set of plausible solutions.

Naturally, the authors of the approach were not really concerned with the procedural perspective on abductive reasoning, but rather with getting a grasp on the underlying logical foundations, which admittedly, semantic tableaux method provides. What is required in order to turn this basis into a reasonably efficient computational method is, as argued before, a goal-oriented guiding mechanism. This mechanism can be found in the *connection restriction* to the clause tableau proof method, which prohibits certain extension steps in the process of constructing a tableau tree.

Connection Restriction: when expanding a branch, use a clause only if it contains a literal that is complementary to the literal in the current leaf.

The connection refinement, thoroughly discussed in [Hähnle, 2001], originates directly from the results on connection-based theorem proving, and thus implements the idea of restricting the search space to formulas that can contribute to the proof. As stated in the following theorem, the connection tableau method is a complete proof system:

Theorem 3 ($\vdash_{\mathcal{T}_{RC}}$: completeness [Hähnle, 2001, Thm. 4.14], [Hähnle et al., 2004, Thm. 3])**.** *If a finite ground clause set S is unsatisfiable then there is a regular connection tableau proof for S, in which a relevant clause $Cl \in S$ is the first to which a beta rule is applied.*

In fact, the theorem is even a bit stronger and covers yet another refinement, *regularity*, which can be expressed as follows:

Regularity Restriction: when expanding a branch, use a clause only if it does not contain a literal that already occurs on the branch.

We can now summarize the two introduced modifications with a formal definition of a regular connection tableau.

Definition 9 (Regular connection tableau [Hähnle, 2001, Def. 4.7, 4.11])**.** A *regular connection tableau* is a clause tableau in which every inner node L (the root to be excluded) has $\overline{L}$ as one of its immediate successors and none of the nodes L has L as any of its predecessors.

Notice, that since both refinements can be actually seen as restrictions to the choice of extension steps, it follows that every proper $\vdash_{\mathcal{T}_{RC}}$ tableau is at the same time a proper $\vdash_{\mathcal{T}}$ tableau. Hence, the following property, which automatically establishes soundness of $\vdash_{\mathcal{T}_{RC}}$, holds.

Proposition 3 ($\vdash_{\mathcal{T}_{RC}}$: soundness)**.** $\vdash_{\mathcal{T}_{RC}} \subseteq \vdash_{\mathcal{T}}$

The cost for both improvements and a significant difference with respect to the general tableau procedure is the loss of *proof confluency*. Notice, that the completeness theorem guarantees the existence of a closed tableau tree, but does not say that there is a deterministic algorithm for constructing one. This means that some proofs might get stuck and backtracking is necessary. Having this consequence in mind, we will appropriately adjust the selection strategy for plain solutions to abductive problems, and once again redefine the whole setting.

Definition 10 ($\vdash_{\mathcal{T}_{RC}}$-abduction)**.** Let $\langle \mathcal{K}, \varphi \rangle$ be an abductive problem. A formula α is a *plain solution* to $\langle \mathcal{K}, \varphi \rangle$ iff there exists a regular connection tableau for $\mathcal{K} \cup \{\neg\varphi\}$, initiated by (beta expansion) of $\neg\varphi$, such that $Cl_{\neg\alpha} = \Sigma_T$, where Σ_T is the collection of leaves from open branches of T. Moreover, we call α *consistent* iff $\mathcal{K} \nvdash_{\mathcal{T}^*_{RC}} \neg\alpha$.

Notice, that this time the search is conducted by going through all possible $\vdash_{\mathcal{T}_{RC}}$ tableau trees, as every tableau tree designates, according to the definition, exactly one solution. The relevance and minimality conditions are both to be verified using $\subseteq$-ordering comparison: between solutions and the query, in the first case, and pairwise between solutions, in the second. The reason for which it is not possible to use the previous, $\vdash_{\mathcal{T}}$ characterization of minimality, is that we no longer consider tableaux representing the whole knowledge, but only its fractions. Hence, a solution that is $\vdash_{\mathcal{T}}$-minimal on one tree can turn out non-minimal on another. The inference denoted as $\vdash_{\mathcal{T}^*_{RC}}$, serving here for goal-oriented consistency checking, is a slightly relaxed variant of the regular connection tableau method, permitting the use of the so-called restart rule, which brings back proof confluency. This procedure will be discussed in Section 3.4, and as for now let us take it for granted that it is also a sound and complete proof method for the task it is intended for.

Given $\mathcal{K}$ is finite, there is a finite number of connection tableaux that can be constructed, hence termination of the procedure is guaranteed.

Before proving the validity of the method and closing the section, let us illustrate it with a small example, based again on the Koala problem. Let then $\langle \mathcal{K}, \mathsf{anxious\_Koala} \rangle$ be the abductive problem, where $\mathcal{K}$ is specified as in the beginning of the chapter. The tableau trees in Figure 3.1 represent the total search space covered while solving the problem. The resulting set of plain solutions consists of four different conjunctive formulas, whose negations are leaves on the open branches of the four tableaux. The first solution is clearly irrelevant, while the three others can be submitted to the consistency checking procedure.

The proof of soundness and completeness of $\vdash_{\mathcal{T}_{RC}}$-abduction is fairly simple and follows immediately from the definitions and theorems introduced so far.

Theorem 4 ($\vdash_{\mathcal{T}_{RC}}$-abduction: soundness and completeness)**.** *A formula α is a plain, consistent and minimal solution to the problem $\langle \mathcal{K}, \varphi \rangle$* iff *$\alpha$ is a $\vdash_{\mathcal{TRC}}$ plain, consistent and minimal solution to $\langle \mathcal{K}, \varphi \rangle$.*

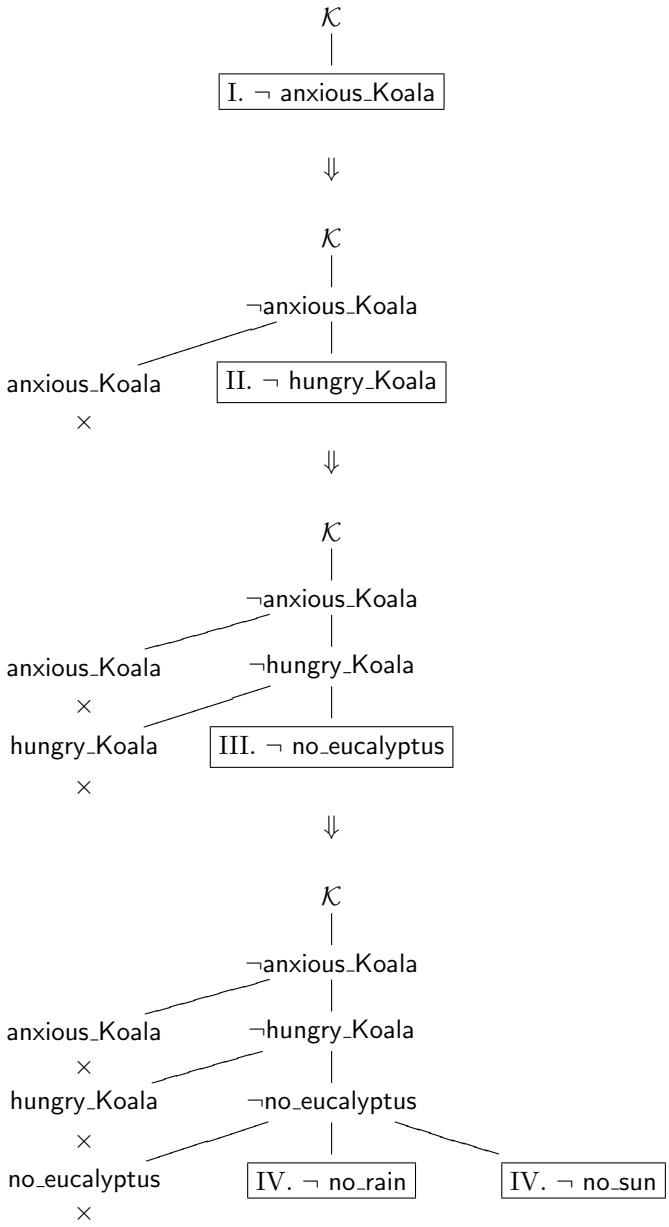


Figure 3.1: $\vdash_{\mathcal{T}_{RC}}$-abduction search space for the Koala problem.

Proof. ($\Rightarrow$) Let α be a plain, consistent and minimal solution to $\langle \mathcal{K}, \varphi \rangle$. Then, by Lemma 1 there exists a mu subset of $\mathcal{K} \cup Cn_\alpha \cup \{\neg\varphi\}$ subsuming $Cn_\alpha \cup \{\neg\varphi\}$.

Hence, $\neg\varphi$ is relevant in $\mathcal{K} \cup Cn_\alpha \cup \{\neg\varphi\}$, and hence can be used as an initial clause. By Theorem 3 there has to exist a tableau proof of $\mathcal{K} \cup Cn_\alpha \vdash_{\mathcal{T}_{RC}} \varphi$. Imagine then, that elements of Cn_α are in fact placed in the root of the tree along $\mathcal{K}$ and that appropriate proof T is constructed. Notice, that every conjunct of α has to be complemented on one of the branches, or otherwise the proof would not succeed, as α is minimal. Imagine now we remove elements of Cn_α from the root: we have a proper, open $\vdash_{\mathcal{T}_{RC}}$-tree left, whose leaves comprise set Σ_T such that $Cl_{\neg\alpha} = \Sigma_T$. But then such tree can be found by an exhaustive search through all possible regular connection tableaux for $\langle \mathcal{K}, \varphi \rangle$.

($\Leftarrow$) This direction follows immediately from soundness of $\vdash_{\mathcal{T}}$-plain abduction (Theorem 2) (notice that every $\vdash_{\mathcal{T}_{RC}}$-plain solution is a sound $\vdash_{\mathcal{T}}$-plain solution); soundness of $\vdash_{\mathcal{T}^*_{\mathcal{RC}}}$ for consistency checking; and finally Proposition 1, guaranteeing a valid minimality verification procedure, knowing that among verified solutions there are all that are minimal (shown in the first part of the proof). □

3.3 Resolution-based approach

The mechanism underlying the connection tableaux method falls very close to *resolution* — another well-known theorem proving technique. Resolution also fully exploits the benefits of a connection-driven search for a proof. The attractiveness of the method stems from the simplicity of its two inference rules, presented in Table 3.3.

$$\frac{Cl_1 \cup \{L\} \quad Cl_2 \cup \{\overline{L}\}}{Cl_1 \cup Cl_2} \qquad \frac{Cl \cup \{L, L\}}{Cl \cup \{L\}}$$

Binary resolution **Factoring**

Table 3.3: Resolution rules.

The rules operate on formulas provided in the form of clauses, but unlike before, clauses should be represented as multisets of literals, i.e. it is allowed for a clause to contain the same literal more than once. A *resolution deduction* of a formula φ from a set of clauses $\mathcal{K}$ denoted as $\mathcal{K} \vdash_{\mathcal{R}} \varphi$, is a derivation of an empty clause $\bot$ from $\mathcal{K} \cup \{\neg\varphi\}$ using the two inference rules. On every application of a rule, a new clause — a *resolvent* or a *factor* — is generated and added to the set of all clauses. The inference halts when none of the rules can be applied anymore. In such case we say that the resulting set of clauses is *saturated*.

Comparably to semantic tableaux, resolution has a longer tradition of applications to modeling abductive reasoning, predominantly as the inference engine employed for running logic programs, for instance in PROLOG. In the context of abductive logic programming, the background theory of an abductive problem is represented as a set of Horn clauses, typically with some of the predicates marked as *abducibles* (for instance using negation-as-failure), i.e. as potential

components of plain solutions (cf. [Kakas et al., 1992]). The inference mechanism is based on linear resolution: a refinement requiring a resolution proof to be linearly ordered, i.e. in such a way that each inference step uses the most recently derived clause (cf. [Hähnle et al., 2004]). A formula is a plain solution to an abductive problem if it consists only of abducibles required for completing the refutation of the query.

Outside of logic programming, there has been at least one attempt of devising a resolution-based abductive procedure, geared for deriving basic solutions to abductive problems (see Definition 5). The method, reported in [Paul, 1993], is founded on linear resolution and searches for all dead-ends of possible linear proofs (a selection criterion that could be roughly compared to considering only saturated connection tableau trees). Alas, the method fails to be logically complete.

Clearly, both settings are too restrictive to be useful from our perspective. What we are interested in here is a generic approach admitting any propositional theories and selecting all solutions that fall under the definition introduced in Section 2.2.

By analogy to tableaux, it comes as a reasonable presumption that since resolution is a refutation complete proof system, there should be a way of enforcing refutation of $\mathcal{K} \cup \{\neg\varphi\}$ for an abductive problem $\langle \mathcal{K}, \varphi \rangle$, by adding some formula to the proof, whenever the refutation cannot succeed naturally. Such a formula will be clearly a plain solution to the problem. The presumption is correct, and the formulas that can facilitate the refutation are the negations of the clauses that are present in the clause set at any stage of the run of resolution inference. Based on this premise, we could start by developing a general resolution-based framework, which similarly to the general tableau approach, would turn out extremely inefficient, as usually not all formulas from the knowledge base are relevant for solving a particular abductive problem. Since the literature does not address such an approach as well, we will skip that work, and instead immediately propose an appropriate refinement, which grants a much more computationally manageable procedure. The applied strategy is known as *set-of-support* and defined as follows:

Definition 11 (Set-of-support [Loveland, 1978, Def. 3.2.1.])**.** A deduction of clause Cl from a set S of clauses is a *deduction with set-of-support* $T \subseteq S$ iff every resolvent of the deduction has at least one parent that is (a factor of) a resolvent or (a factor of) a member of T.

The strategy imposes additional constraints on the application of the inference rules, which could now be represented via revised schemes presented in Table 3.4. Notice, that binary resolution is restricted only to such pairs of clauses, whose one member belongs to the set-of-support. Additionally we restrict the application of factoring to the clauses from the set-of-support only. Moreover, every inferred clause is added to the set-of-support. As a consequence, if we partition the input set of clauses S into two sets S_0 and T_0, such that T_0 serves as the initial set-of-support, then the run of the procedure can be

$$\frac{Cl_1 \cup \{L\} \in S \quad Cl_2 \cup \{\overline{L}\} \in T}{Cl_1 \cup Cl_2 \in T} \qquad \frac{Cl \cup \{L, L\} \in T}{Cl \cup \{L\} \in T}$$

Binary resolution **Factoring**

Table 3.4: Resolution with set-of-support rules.

represented as a chain of $\vdash_{\mathcal{R}_S}$–inferences corresponding to consecutive updates of T_0, as illustrated below.

$$S_0 \cup T_0 \vdash_{\mathcal{R}_S} S_0 \cup T_1 \vdash_{\mathcal{R}_S} \ldots \vdash_{\mathcal{R}_S} S_0 \cup T_i \vdash_{\mathcal{R}_S} \ldots \vdash_{\mathcal{R}_S} S_0 \cup T_n$$

Soundness of resolution with set-of-support follows immediately from acknowledging that the method is merely a more restrictive variant of the general resolution procedure, whereas completeness is established by Theorem 5.

Proposition 4 ($\vdash_{\mathcal{R}_S}$: soundness)**.** $\vdash_{\mathcal{R}_S} \subseteq \vdash_{\mathcal{R}}$

Theorem 5 ($\vdash_{\mathcal{R}_S}$: completeness [Loveland, 1978, Thm. 3.2.2.])**.** *If S is an unsatisfiable set of clauses and $T \subseteq S$ such that $S - T$ is satisfiable then there exists a refutation of S with set-of-support T.*

Furthermore, it is possible to incorporate other useful refinements that considerably improve performance of the procedure. The following two are proven to be compatible with the set-of-support strategy [Loveland, 1978].

Subsumption At any stage S_i of the run of resolution, for every two clauses $Cl_1, Cl_2 \in S_i$, if $Cl_1 \subseteq Cl_2$ then Cl_2 can be removed from S_i.

Tautology deletion At any stage S_i of the run of resolution, for every clause $Cl \in S_i$, if Cl is a tautology, i.e. there is a pair of complementary literals in Cl, then Cl can be removed from S_i.

Naturally, it has to be ensured, that clauses once removed, are not generated again via the same sequences of inference steps, and more generally, that every resolution step is applied only once to a pair of clauses. If this is the case, then given a finite propositional input, $\vdash_{\mathcal{R}_S}$ is guaranteed to reach the saturation point of the set-of-support and thus terminate after finite number of inferences, as every resolvent contains two occurrences of literals less (relatively to the union of its parents) that could be possibly used in the consecutive inferences [Loveland, 1978]. Note, that this saturated set-of-support will be of prime importance for abductive reasoning.

We are ready now to define a suitable setting for finding all worthwhile solutions to an abductive problem by means of the set-of-support resolution procedure.

Definition 12 ($\vdash_{\mathcal{R}_S}$-abduction)**.** Let $\langle \mathcal{K}, \varphi \rangle$ be an abductive problem, and T_n the saturated set-of-support in a run of resolution for $\mathcal{K} \vdash_{\mathcal{R}_S} \varphi$ with $T_0 = \{\neg\varphi\}$

as the initial set-of-support. A formula α is a *plain* and *minimal solution* to $\langle \mathcal{K}, \varphi \rangle$ iff there exists a clause $Cl \in T_n$ such that $Cl_{\neg\alpha} = Cl$. Moreover, we call α *consistent* iff $\mathcal{K} \nvdash_{\mathcal{R}_S} \neg\alpha$ with Cn_α as the initial set-of-support.

Interestingly, due to the subsumption refinement, what we obtain as the direct output of the procedure is the set off all plain and minimal solutions, which only have to be verified by means of the consistency checking procedure. This procedure can be also effectively based on resolution with set-of-support. We provide formal justification for this claim in Section 3.4, although it should be already fairly clear.

We summarize the section with a representation of the Koala problem in the $\vdash_{\mathcal{R}_S}$-abduction framework, depicted in Figure 3.2, and a proof of soundness and completeness of the procedure.

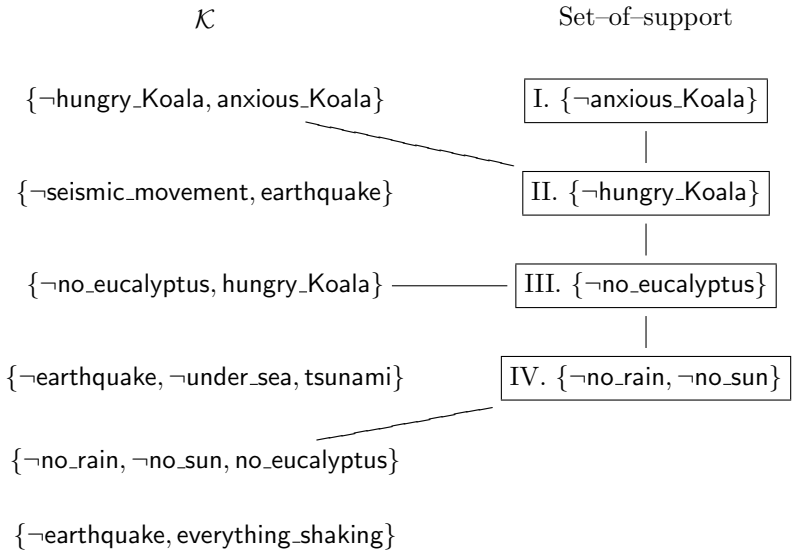


Figure 3.2: $\vdash_{\mathcal{R}}$-abduction search space for the Koala problem.

Theorem 6 ($\vdash_{\mathcal{R}_S}$-abduction: soundness and completeness)**.** *A formula α is a plain, consistent and minimal solution to the problem $\langle \mathcal{K}, \varphi \rangle$* iff *$\alpha$ is a $\vdash_{\mathcal{R}_S}$ plain, consistent and minimal solution to $\langle \mathcal{K}, \varphi \rangle$.*

Proof. ($\Rightarrow$) Let α be a plain, consistent and minimal solution to $\langle \mathcal{K}, \varphi \rangle$, and let T_n be the saturated set-of-support in a run of resolution for $\mathcal{K} \vdash_{\mathcal{R}_S} \varphi$ with $T_0 = \{\neg\varphi\}$ as the initial set-of-support. First, we note that since α is a consistent solution, then it cannot contain complementary literals nor it can entail $\bot$ with $\mathcal{K}$ alone. Having acknowledged that, we prove by induction on the cardinality of $Cl_{\neg\alpha}$ that $Cl_{\neg\alpha} \in T_n$.

Consider $|Cl_{\neg\alpha}| = 0$. This is the limit case, when no abduction is actually needed, as the query already follows from the knowledge base. The only minimal solution is therefore the empty conjunctive formula $\alpha = \top$. But then completeness of resolution guarantees that the empty clause will be derived in T_n, hence $Cl_{\neg\alpha}$ is indeed in T_n.

Assume that for any $Cl_{\neg\alpha}$ and some k, such that $|Cl_{\neg\alpha}| = k$, $Cl_{\neg\alpha}$ is in T_n and consider $|Cl_{\neg\alpha}| = k+1$. Let β be a conjunctive formula and L a literal such that $Cl_{\neg\beta} = Cl_{\neg\alpha} \setminus \{\overline{L}\}$. Imagine that after T_n has been obtained, the unit clause $\{L\}$ is added to the knowledge base and the inference is resumed. Clearly, β becomes a minimal solution to $\langle \mathcal{K} \cup \{L\}, \varphi \rangle$ and $\mathcal{K} \cup \{L\}$ remains satisfiable, hence by the inductive assumption $Cl_{\neg\beta}$ has to occur in the saturated set-of-support. We argue that derivation of $Cl_{\neg\beta}$ has to succeed in a single resolution step by resolving $\{L\}$ against $Cl_{\neg\beta} \cup \{\overline{L}\}$. If $\{L\}$ was not to be used at all, then α should not be a minimal solution, which contradicts the assumption, as a shorter conjunction β solves the problem. If the proof was supposed to take more than one step, then there would have to be a clause $Cl \cup \{\overline{L}\} \in T_n$ resolved against $\{L\}$, such that Cl is resolved against other clauses. But this would mean that T_n was not saturated at the first place, as the assumption states, since Cl could have been resolved against other clauses before $\{L\}$ was added to the knowledge base. Therefore $Cl_{\neg\alpha}$ is indeed in T_n, which concludes the induction proof.

Finally, if α is a consistent solution, then it will be the case that $\mathcal{K} \nvdash_{\mathcal{R}_S} \neg\alpha$ with Cn_α as the initial set-of-support.

($\Leftarrow$) This direction is again straightforward. For each α such that $Cl_{\neg\alpha}$ is clearly a plain solution as adding Cn_α to the knowledge base would allow for resolving $Cl_{\neg\alpha}$ against all unit clauses from Cn_α and deriving the empty clause, which given completeness of $\vdash_{\mathcal{R}_S}$ would signify entailment of the query. Further, since we know by the first part of the proof that all minimal solutions are in fact in T_n, it follows that α is a minimal solutions, as the subsumption refinement corresponds directly to deletion of non-minimal solutions according to $\subseteq$-ordering comparison between solutions. Finally, $\mathcal{K} \nvdash_{\mathcal{R}_S} \neg\alpha$ guarantees that α is a consistent solution. □

3.4 Consistency checking and CNF transformation

The methods discussed in the previous sections provide sufficient means for finding all plain and minimal solutions to any abductive problem $\langle \mathcal{K}, \varphi \rangle$. In order to accomplish the procedural specification of abduction, we have to choose efficient decision procedures for determining consistency of solutions. For this purpose we can use basically the same reasoning techniques, thus being faithful to the policy of keeping the whole process goal-oriented.

By initial condition $\mathcal{K} \not\models \{\neg\varphi\}$ (Def. 3), it follows that $\mathcal{K}$ is satisfiable. This consequence allows us to restrict the search space of possible inconsistencies only to those parts of the knowledge base that are connected to particular solutions being verified, as only a newly added formula can entail contradiction in otherwise consistent set of formulas. As signaled before, for any solution α to $\langle \mathcal{K}, \varphi \rangle$ the following two procedures do exactly the job.

1. α is consistent *iff* $\mathcal{K} \nvdash_{\mathcal{R}_S} \neg\alpha$ with Cn_α as the initial set-of-support.
2. α is consistent *iff* $\mathcal{K} \nvdash_{\mathcal{T}^*_{RC}} \neg\alpha$ with Cn_α as the initial set of literals used on the tree.

In the first case we use resolution over $\mathcal{K} \cup Cn_\alpha$ with set-of-support Cn_α. Since $\mathcal{K}$ is guaranteed to be satisfiable, therefore due to soundness and completeness of $\vdash_{\mathcal{R}_S}$ (Prop. 4, Thm. 5) the empty clause will be derived if and only if $\mathcal{K} \cup Cn_\alpha$ is unsatisfiable, which demonstrates that $\mathcal{K} \cup \{\alpha\} \not\models \bot$.

The second case involves only a slight modification of the regular connection tableau procedure. Since now the goal is not to find all possible tableau proofs, but rather to determine as fast as possible whether there exists a satisfiable model for $\mathcal{K} \cup Cn_\alpha$, therefore the loss of proof confluency becomes bothersome. A simple way of regaining it is to introduce *restart rule*, which enables to resume expansion of a branch whenever the proof gets stuck. In such situations it is permitted to use a clause connected to any node above the leaf, up to literals from Cn_α. Assuming $\mathcal{K} \cup Cn_\alpha$ is inconsistent, one of the literals in Cn_α is clearly relevant (i.e. it belongs to a mu subset of $\mathcal{K} \cup Cn_\alpha$), hence what is eventually obtained is in fact just a more compact search through all regular connection tableaux for $\mathcal{K} \vdash_{\mathcal{T}^*_{RC}} \neg\alpha$, modulo the initial literal. Even the regularity restriction, which might seem suspicious in this setting at the first glance, as it blocks the use of some formulas in supposedly different connection proofs, cannot affect completeness. Notice, that expanding a branch with a formula that contains a literal already used on the branch leads to unnecessary branching, as one of the new branches remains a proper subset of the others. Soundness of $\vdash_{\mathcal{T}^*_{RC}}$ is straightforwardly inherited from soundness of the regular connection tableaux method. As a result, we conclude that $\mathcal{K} \nvdash_{\mathcal{T}^*_{RC}} \neg\alpha$ proves that $\mathcal{K} \cup \{\alpha\} \not\models \bot$.

The process of searching for plain solutions and verifying their consistency can be a subject to further, interesting optimizations. The simplest strategy of consistency checking, applicable only to $\vdash_{\mathcal{T}^*_{RC}}$, is a depth-first search for the first satisfiable model and halting the procedure on succeeding. In some scenarios, however, it might turn out more efficient to further exploit the properties of a connected-driven generation of solutions and elaborate on the following pattern: if α is a plain solution to φ then all consequences of $\mathcal{K} \cup \{\varphi\}$ are also consequences of $\mathcal{K} \cup \{\alpha\}$. Therefore, if $\mathcal{K} \cup \{\varphi\}$ is unsatisfiable then it is totally futile to spend time on verifying consistency of α. To generalize this observation consider again the Koala problem.

$$\ldots \rightarrow \mathsf{no\_eucalyptus} \rightarrow \mathsf{hungry\_Koala} \rightarrow \mathsf{anxious\_Koala}$$

Notice, that going backwards from the initial goal anxious_Koala to more specific solutions, we obtain some useful information about the entailment ordering of selected formulas. Clearly hungry_Koala entails anxious_Koala, no_eucalyptus entails hungry_Koala and so on. This suggest a more sophisticated approach to consistency checking. Instead of starting a search for a satisfiable model of $\mathcal{K} \cup \{\mathsf{hungry\_Koala}\}$ from scratch, it might be easier to first generate all minimal models for $\mathcal{K} \cup \{\mathsf{anxious\_Koala}\}$ and then "borrow" them as an indispensable foundation for the minimal models of $\mathcal{K} \cup \{\mathsf{hungry\_Koala}\}$; those, in turn, can be reused later, while generating models of $\mathcal{K} \cup \{\mathsf{no\_eucalyptus}\}$. The notion and the representation of a minimal model is relative to the method of consistency checking. In our case it might be the set of saturated branches of a $\vdash_{\mathcal{T}^*_{RC}}$ tableau, or the saturated set-of-support of $\vdash_{\mathcal{R}_S}$, both generated according to the provisions specified above.

Naturally, we do not always move along implication chains while abducing solutions, therefore a more cautious strategy is necessary. One approach we have investigated, suited for the resolution setting, is based on maintaining an entailment tree, generated in parallel to the search for solutions, whose nodes are solutions ordered in such a way, that every node is entailed by all its successors. Moreover, each node is labeled with the set of all minimal models $\mathfrak{M}$ of the corresponding solution given the knowledge base. The root of the entailment tree for $\langle \mathcal{K}, \varphi \rangle$ is fixed to $\langle \varphi, \mathfrak{M}(\varphi) \rangle$, where $\mathfrak{M}(\varphi)$ is the only set of models created without any partial input. For every other node α, $\mathfrak{M}(\alpha)$ is computed on the basis of the models of the immediate predecessor of α. The tree is constructed according to the following rules. For every clause $Cl_{\neg\alpha}$ from set-of-support T:

1. if $Cl_{\neg\alpha}$ is a factor of $Cl_{\neg\beta} \in T$ then α replaces β on the tree;
2. if $Cl_{\neg\alpha}$ is a resolvent of $Cl \in \mathcal{K}$ and $Cl_{\neg\beta} \in T$ then α is an immediate successor of β;
3. if $Cl_{\neg\alpha}$ is a resolvent of $Cl_{\neg\beta} \in T$ and $Cl_{\neg\gamma} \in T$ then α is an immediate successor of the nearest common predecessor of β and γ.

It is easy to show that the tree indeed preserves a sound representation of the entailment ordering of the solutions. The first point is obvious. In the second one notice, that since any clause $Cl_{\neg\gamma} \cup \{L\} \in \mathcal{K}$ is satisfied, then $\mathcal{K} \cup Cn_\gamma \vDash L$. Therefore for any other clause $Cl_{\neg\beta} \cup \{\overline{L}\} \in T$, against which it might be resolved, it holds that $\mathcal{K} \cup Cn_\beta \cup Cn_\gamma \vDash Cn_\beta \cup \{L\}$. In the third case, observe that $\alpha \vDash \beta \vee \gamma$. But what is entailed by the disjunction of β and γ corresponds to all common predecessors of β and γ on the entailment tree, of which the nearest one contains maximum information.

An analogous solution should also be possible for the tableau setting, with the only difference, that the entailment relationships would have to be decided on the basis of the structure of a tableau tree.

Obviously, the gain in time efficiency of consistency checking, offered by such an approach, is traded off against space required for storing all minimal models.

Whether this trade-off is well balanced depends predominantly on the intended use case and the logical structure of a knowledge base.

The last, still quite general improvement, which is going to be suggested, concerns transformation of a knowledge base into Conjunctive Normal Form. Naturally, a thorough preprocessing of the input knowledge is not optimal if only small part of it can be actually useful for solving a given abductive problem. Once again, however, it is possible to render the procedure goal-dependent. Since both types of search that constitute the process of abductive reasoning, i.e. searching for solutions, and searching for their consequences, rely on existence of connections, therefore the same criterion can be engaged for driving the transformation scheme. Given current state of computation specified by a search for the connection to a literal L, a propositional formula in Negation Normal Form might be applicable, i.e. it is reducible to possibly relevant clauses, only if it contains $\overline{L}$. Going one step back, we can determine whether any formula can be applicable by checking if it contains an atomic proposition from L, and only then transform it to NNF. The output of the transformation should replace the original formula, in order to avoid repeating the process on another request for the formula.

3.5 Final remarks

It should be clear by now, that what is promoted in this chapter is a holistic view on all the reasoning tasks comprising the process of solving an abductive problem. It is therefore not one or another reasoning technique that essentially matters, but rather the whole reasoning architecture, which should be handling the process in a goal-oriented manner and optimally reusing partial results obtained on the way. The design described above could be roughly depicted as in Figure 3.3.

As a part of the research we have implemented and tested a propositional reasoner for solving abductive problems. The implementation has been written in Java Expert System Shell (JESS), a rule engine based on the Rete algorithm.[2] The reasoner, presented in more detail in the Appendix, consists of the resolution-based algorithm for finding abductive solutions and the tableaux-based method of checking their consistency. Such a hybrid composition proves a nice, modular character of the reasoning architecture described in the chapter.

In the resolution part, we have merged factoring and binary resolution rules together, whereas the tableau procedure has been augmented with a mechanism for detecting and removing non-minimal models, at the earliest possible stage. The search for connections is efficiently performed by underlying, pattern-matching Rete algorithm. In the course of solving a problem an entailment tree is maintained, along the lines explained in Section 3.4. From the experience of using the reasoner it follows that the biggest share of time is consumed not on finding plain, minimal solutions, but on checking their consistency with the knowledge base. Hence, the most significant improvement to the performance of

[2] `http://herzberg.ca.sandia.gov/`

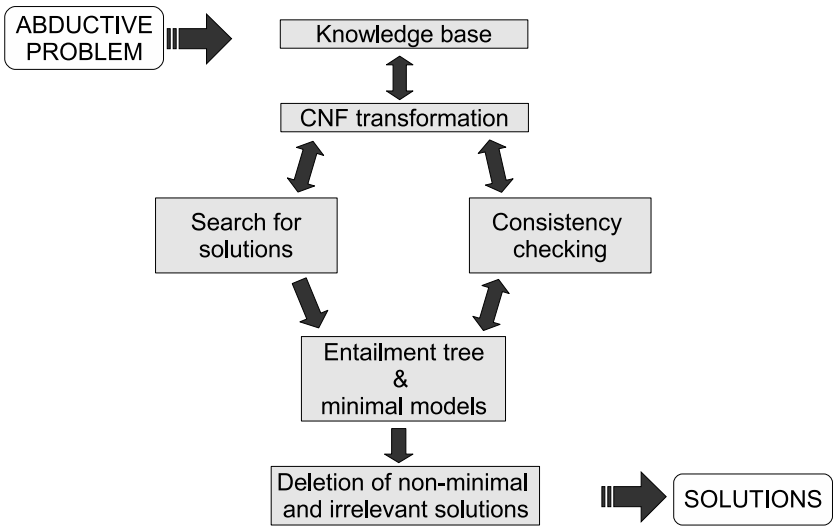


Figure 3.3: The reasoning architecture for solving abductive problems.

reasoning is due to the strategy of reusing models obtained from checking consistency of other solutions. Also, the reasoner performs better on average when search for solutions follows breadth-first strategy, whereas check of consistency depth-first. Breadth-first strategy in solving a problem has also this benefit from the perspective of potential real-life applications, that it leads to finding $\mathcal{K}$-minimal (most general) solutions first. Based on that, it could be possible to engage the user in pruning undesired general solutions early in the process, thus considerably reducing the space of possible specific answers. Another way of making the procedure interactive could be oriented towards optimal reduction of information entropy of the space of solutions. Given all minimal models of different solutions, we can pick certain literals, which (almost) proportionally partition the set of current solutions into ones that entail the literal and ones that entail its complement. Asking the user for providing additional information on the logical value of that literal would cut down the number of plausible solutions by half.

Naturally, there are many other forms of optimizations that could surely benefit the architecture and push the performance of reasoning on a much higher level. One of them, for instance, includes ordering restrictions on possible inference steps. Orderings are very often key refinements in modern theorem provers as their impact on the performance can be substantial. Observe that both abductive procedures described in this chapter can generate exponentially many redundant solutions, simply because the same outcomes are obtained via permutations of the same inference steps (see Figures 3.4 and 3.5). Typically, ordering restrictions are specified with respect to the signature of the language in which the knowledge base is expressed (cf. [Hähnle, 2001,

Bachmair and Ganzinger, 2001]), in such a way that some propositions cannot be connected to the proof unless others are before. While this approach is well developed for general theorem proving, it would have to be essentially revised to suit the settings, like presented here, in which the flow of a proof is partly determined by context.

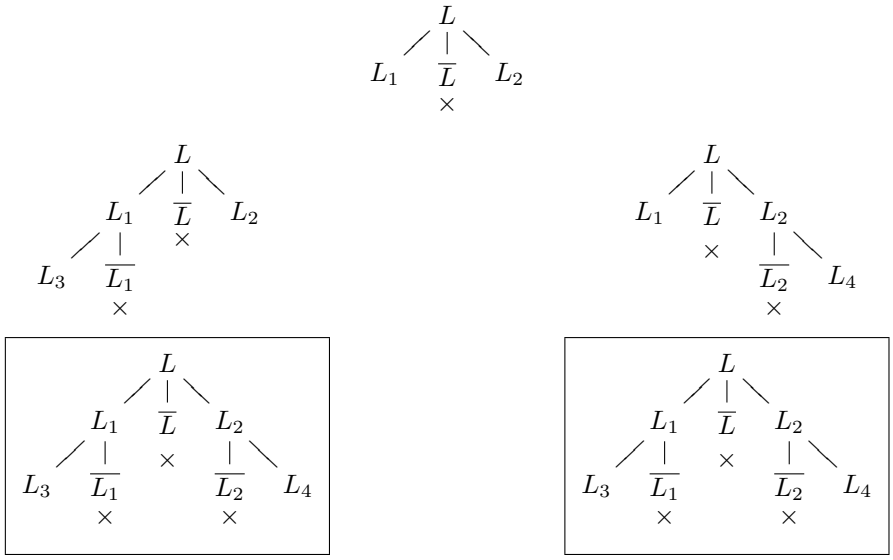


Figure 3.4: Redundancy in $\vdash_{\mathcal{T}_{RC}}$-abduction.

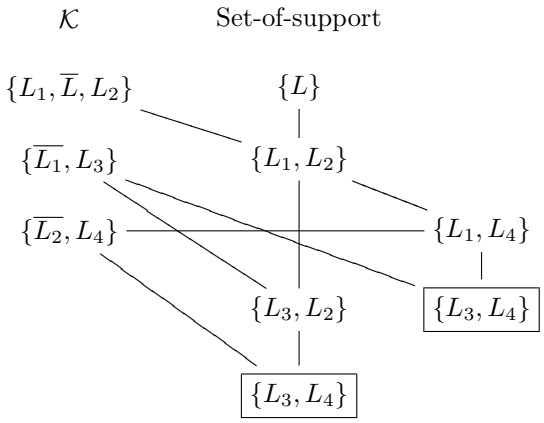


Figure 3.5: Redundancy in $\vdash_{\mathcal{R}}$-abduction.

Nevertheless, we are not going to discuss other possible refinements within the scope of this work anymore. Instead, in the following chapters, we will try to adapt the essential parts of the reasoning framework to Description Logic, thus fundamentally extending the expressiveness of admissible knowledge bases, and consequently, the complexity of abductive problems that can be solved along the same principles.

Chapter 4

Abduction in Description Logic

In the previous chapter we have elaborated on the design principles for goal-oriented automated abductive reasoners. We have argued and demonstrated that connection-driven mechanism of searching for abductive solutions, and also their consecutive assessment with respect to consistency, offers a much better control over the process of reasoning and can significantly reduce the computational effort required for solving a problem. Eventually, we have arrived at two procedures, based on tableaux and resolution techniques, which properly applied, become sound and complete goal-oriented calculi for finding all minimal and consistent solutions to an abductive problem.

Nonetheless, a fine reasoning model cannot be of much use in real-life applications, unless the knowledge representation formalism it is tailored for, is sufficiently expressive to handle problems of practical importance. So far we have only considered knowledge bases and abductive problems described in propositional logic. The expressive limits of this formalism are obvious, and nobody could seriously treat it nowadays as an adequate language for representing and reasoning with knowledge about virtually any domain amenable to machine-accessible modeling. Our ambitious objective, therefore, is to go beyond those constraints and lay down formal foundations for exactly the kind of *practical reasoning tools*, that could be useful in application to knowledge systems meeting state-of-the-art standards of the field of Knowledge Representation. To this end, we will try to extend the introduced framework to Description Logic — a leading paradigm of logic-based KR.

In this chapter we provide a short introduction to the syntax and semantics of DL languages and define ABox abduction, which is of central interest to us. In Section 4.3 we discuss several problematic issues arising from first attempts of adapting the abductive framework to the new formalism and point to more promising directions, which will be fully explored in the next chapter.

4.1 Description Logics

Description Logics are a family of logical languages intended particularly for representing knowledge about a domain of application. They descend from such KR formalisms as semantic networks and frames, both of high popularity in AI of 70's and 80's [Nardi and Brachman, 2003]. From that, mostly cognitive science-flavored root, they inherit a concept-oriented layout of the syntax and a characteristic partitioning of the expressible formulas into the *terminological* and the *assertional* fragment. On the other hand, DLs are grounded on a proper, model-theoretic semantics and remain decidable logics — features not achievable for the predecessors. Due to these properties DLs gained a lot of interest in the beginning of the last decade in the KR community, and have been under intensive development since then.

Recently, it has been decided that DLs are going to form the logical foundation of the Semantic Web — the future incarnation of the Internet. More specifically, DLs have been employed to underpin the Web Ontology Language, a language for expressing ontologies and knowledge about web resources [Horrocks et al., 2003]. In this new context, DL has turned out very successful and currently a large research community, in cooperation with the W3C standardization group and OWL's end-users, works on designing more expressive DLs and efficient tools providing a variety of reasoning services in order to to satisfy ever growing application demands.

A signature $\Sigma = (N_I, N_C, N_R)$ of a DL knowledge base consists of three sets of names: for individuals, atomic concepts and simple roles, respectively [Baader and Nutt, 2003]. The semantics is given by an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty domain of individuals and $\cdot^{\mathcal{I}}$ is an interpretation function defining the meaning of the vocabulary as follows:

- $\cdot^{\mathcal{I}} : N_I \mapsto \Delta^{\mathcal{I}}$ for individual names,
- $\cdot^{\mathcal{I}} : N_C \mapsto \wp(\Delta^{\mathcal{I}})$ for concept names,
- $\cdot^{\mathcal{I}} : N_R \mapsto \wp(\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}})$ for role names.

By default, we also treat $\top$ and $\bot$, where $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$, $\bot^{\mathcal{I}} = \emptyset$, as fixed components of a DL language. The reminder of the semantics is defined inductively on the construction rules for complex expressions.

Every DL language is characterized by a set of constructors available for building complex formulas. This set signifies the expressive power of a language, denoted by conventional abbreviations. Table 4.1 presents the list of constructors of the *basic attributive language* ($\mathcal{AL}$) with *complex concept negation* ($\mathcal{C}$), which will be the main reference for our considerations. All of them are concept constructors, i.e. for arbitrary concepts C and D, and a role r, every of the resulting expressions is another concept, or more precisely: a *concept description*.

Some of the possible expressive extensions to $\mathcal{ALC}$, which will be also mentioned in the report, include *inverse roles* ($\mathcal{I}$), *nominals* ($\mathcal{O}$) and *cardinality*

Constructor	Syntax	Semantics
concept negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
concept intersection	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
concept union	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{x \mid \exists_y(\langle x, y\rangle \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}})\}$
universal restriction	$\forall r.C$	$\{x \mid \forall_y(\langle x, y\rangle \in r^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}})\}$

Table 4.1: $\mathcal{ALC}$ constructors.

restrictions ($\mathcal{N}$), ($\mathcal{Q}$). The syntax and semantics of these constructors are provided in Table 4.2, for an arbitrary role r, individual a, concept C and a natural number n. Note, that the first one is a role constructor, whereas the others are again complex concept constructors.

Constructor	Syntax	Semantics
role inverse	r^-	$\{\langle y, x\rangle \mid \langle x, y\rangle \in r^{\mathcal{I}}\}$
nominal	$\{a\}$	$\{a^{\mathcal{I}}\}$
minimum cardinality	$\geq n\ r$	$\{x \mid \ \lvert\{y \mid \langle x, y\rangle \in r^{\mathcal{I}}\}\rvert \geq n\}$
maximum cardinality	$\leq n\ r$	$\{x \mid \ \lvert\{y \mid \langle x, y\rangle \in r^{\mathcal{I}}\}\rvert \leq n\}$
qualified minimum cardinality	$\geq n\ r\ C$	$\{x \mid \ \lvert\{y \mid \langle x, y\rangle \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}\rvert \geq n\}$
qualified maximum cardinality	$\leq n\ r\ C$	$\{x \mid \ \lvert\{y \mid \langle x, y\rangle \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}\rvert \leq n\}$

Table 4.2: $\mathcal{I}$, $\mathcal{O}$, $\mathcal{N}$, $\mathcal{Q}$ constructors.

A DL knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ consists of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$. All formulas included in $\mathcal{K}$ are denoted as TBox or ABox axioms.

The TBox is a formal representation of the terminological part of the knowledge base, establishing relationships between concepts and roles. In the elementary variant, it comprises a set of *concept definitions* of the form $A \doteq C$, where A is an atomic concept and C an arbitrary concept description. More liberal representation of so-called *general TBoxes*, which will be considered in the reminder of the thesis, is based on *general concept inclusions* (GCI), such as $C \sqsubseteq D$ — corresponding to the implication between arbitrary descriptions C and D. When the inclusion holds in both directions the concepts are said to be *equivalent* $C \equiv D$. Additionally, more expressive languages, allow also *role hierarchies* ($\mathcal{H}$), which follow the notation of GCI. For instance $r \sqsubseteq p$ expresses that r is a subrole of p.[1]

[1]Currently, as the research on languages allowing for more flexibility in defining roles and relationships between them is quickly progressing (such as $\mathcal{SROIQ}$ underlying OWL 2, see [Horrocks et al., 2006]) it is becoming more common to separate GCIs from role-related axioms, and collecting the latter in the RBox, which is acknowledged as the third component of a DL knowledge base.

The ABox of a knowledge base consists of a set of assertions about individuals, of the form $C(a)$ or $r(a, b)$, where a, b are names of individuals, C a concept description, and r a role. The former states that a is an instance of C, whereas the latter — that individual a is related to b via role r.

The semantics of TBox and ABox axioms is defined in a standard way, presented in Table 4.3. An interpretation $\cdot^{\mathcal{I}}$ *satisfies* an axiom iff the axiom's semantics is respected under $\cdot^{\mathcal{I}}$. An interpretation is a *model* of a knowledge base iff it satisfies all the axioms.

Axiom	**Semantics**
$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$C \equiv D$ $(C \doteq D)$	$C^{\mathcal{I}} = D^{\mathcal{I}}$
$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
$r(a, b)$	$\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in r^{\mathcal{I}}$

Table 4.3: Semantics of DL axioms.

Finally, let us define the notion of *cyclic terminology* in the context of general TBoxes.[2] Let A and B be two atomic concepts. We say that A *directly uses* B in $\mathcal{T}$ iff there is an axiom in $\mathcal{T}$ such that A is on its right-hand side, while B on the left-hand side. We denote the transitive closure of *directly uses* as *uses*, and say that $\mathcal{T}$ is a cyclic terminology iff there is an atomic concept which uses itself in $\mathcal{T}$. Reversely, *acyclic terminologies* are ones that do not contain concepts using themselves.

A serious consequence for reasoning over cyclic terminologies is that any proof procedure has to account for some kind of loop detection, for the termination to be guaranteed. Consider for instance a recurrent example of the definition of a person:

$$\textsc{Person} \equiv \exists \mathsf{hasParent}.\textsc{Person}$$

Clearly, for many reasoning techniques, a naive approach to checking satisfiability of an assertion $\textsc{Person}(\mathsf{John})$ would eventuate in an infinite generation of hasParent-successors of John. For that reason, special mechanisms that allow to retain decidability, known as *blocking rules*, have been carefully devised for DLs of particular expressiveness, and employed in automated reasoners.

Some very important results in the field of DL, both from the theoretical and practical perspective, regard the correspondence of the formalism to other logics. It has been noticed, for instance, that $\mathcal{ALC}$ can be seen as a notational variant of multimodal logic K_n [Schild, 1991, de Rijke, 1998]. The mapping, shown in Table 4.4, identifies individuals with possible worlds, roles with accessibility relations and atomic concepts with atomic propositions. Quantification restrictions correspond to diamond and box modalities. Consequently, DLs can be mapped to a subset of FOL through the well-known *standard translation* in

[2]We thus generalize the definition used in [Baader and Nutt, 2003].

modal logic [Blackburn and van Benthem, 2006], presented in Table 4.5. Under the translation atomic concepts and roles are mapped respectively to unary and binary predicates, whereas individuals are treated as FOL constants.

$\pi^K(A)$	$=$	p_A
$\pi^K(\neg C)$	$=$	$\neg\pi^K(C)$
$\pi^K(C \sqcup D)$	$=$	$\pi^K(C) \vee \pi^K(D)$
$\pi^K(C \sqcap D)$	$=$	$\pi^K(C) \wedge \pi^K(D)$
$\pi^K(\forall r.C)$	$=$	$[R_r]\pi^K(C)$
$\pi^K(\exists r.C)$	$=$	$\langle R_r\rangle\pi^K(C)$

Table 4.4: Correspondence to multimodal logic K_n.

$st_x(A)$	$=$	$P_A(x)$
$st_x(\neg C)$	$=$	$\neg st_x(C)$
$st_x(C \sqcup D)$	$=$	$st_x(C) \vee st_x(D)$
$st_x(C \sqcap D)$	$=$	$st_x(C) \wedge st_x(D)$
$st_x(\forall r.C)$	$=$	$\forall_y(r(x,y) \rightarrow st_y(C))$
$st_x(\exists r.C)$	$=$	$\exists_y(r(x,y) \wedge st_y(C))$

Table 4.5: Standard translation.

The correspondences have facilitated a transfer of many results and methods developed for the other logics to DL, predominantly regarding semantic foundations, computational complexity, and core reasoning techniques. However, as will be also highlighted in the case of abduction, the analogy between the formalisms can fall short. For instance, in modal logics it is not common to consider background theories, i.e. formulas true in every possible world, whereas a DL's TBox is clearly such a theory. In more expressive DLs there are also some constructs (e.g. transitive roles) that cannot be directly expressed in the FOL syntax. We will further elaborate on these issues in Section 4.3, while making first steps towards redefining the abductive framework to DL.

4.2 ABox abduction

Description Logics, with their rich means of expressiveness and epistemically motivated clustering of knowledge bases, provide a wide range of interesting contexts for investigating and applying forms of abductive reasoning. In [Elsenbroich et al., 2006] the authors initiated a discussion on the place for abduction in DL ontologies, and introduced the first, broad classification of the relevant types of abductive problems. Let us shortly restate these distinctions, considering an arbitrary DL knowledge base $\mathcal{K}$:

Concept abduction*:** given a concept C find a concept H such that $\mathcal{K} \vDash H \sqsubseteq C$. A special case of that type of abduction is ***conditionalized concept

abduction, where the goal is to find an H such that $\mathcal{K} \models H \sqcap D \sqsubseteq C$, for some fixed concept D.

***TBox abduction*:** given a GCI $C \sqsubseteq D$ find another GCI $E \sqsubseteq F$ such that $\mathcal{K} \cup \{E \sqsubseteq F\} \models C \sqsubseteq D$.

***ABox abduction*:** given an assertion $\varphi(\overline{a})$ find a set of assertions A such that $\mathcal{K} \cup A \models \varphi(\overline{a})$.

***Knowledge base abduction*:** given an ABox or TBox axiom φ find a set of ABox and/or TBox axioms A such that $\mathcal{K} \cup A \models \varphi$.

So far abduction in DL has been approached only in [Colucci et al., 2004], where a tableaux-based algorithm for conditionalized concept abduction in acyclic $\mathcal{ALN}$ TBoxes has been proposed, in order to support so-called matchmaking tasks. More recently, some attention has been given in [Peraldi et al., 2007] and [Möller and Neumann, 2008] to the problem of facilitating interpretation of visual data using ontologies with DL rules (i.e. Horn clauses based on the signature of a DL knowledge base, satisfying certain additional conditions). The authors devised a simple inference mechanism for ABox abduction over the rule bodies, which suggests ways of enhancing the conceptualization of the data. To our knowledge, no other work has been done directly on the problem.[3]

Clearly, the approaches taken in both cases are quite narrowly scoped, concentrating either on a specific type of DL terminologies (acyclic $\mathcal{ALN}$) or on constructs from beyond DL (DL rules), in the context of particular application scenarios. Thus, both of them are of little value for the task taken in this work, which is to investigate the possibility of designing a uniform procedure for ABox abduction in DL. Even though the main focus will be on DL $\mathcal{ALC}$ we want the approach to be generic, so that its lifting to expressive extensions of $\mathcal{ALC}$ would be possible.

The following is the definition of the formal setting for ABox abduction as it will be understood in the reminder of the report.

Definition 13 (ABox abduction)**.** Let $\mathcal{L}$, $\mathcal{L}_Q$ and $\mathcal{L}_S$ be DLs, $\mathcal{K}$ a knowledge base in $\mathcal{L}$ and Φ a set of ABox assertions in $\mathcal{L}_Q$, such that for every $\varphi \in \Phi$ it holds that $\mathcal{K} \not\models \neg\varphi$. We call tuple $\langle \mathcal{K}, \Phi \rangle$ an *ABox abduction problem.* A set of assertions A in $\mathcal{L}_S$ is a *plain solution* to $\langle \mathcal{K}, \Phi \rangle$ iff $\mathcal{K} \cup A \models \Phi$. Moreover, we call A:

1. *consistent* iff $\mathcal{K} \cup A \not\models \bot$.
2. *minimal* iff for every solution B if $A \models B$ then $B \models A$.
3. *solipsistic* iff for every $\alpha(\overline{a}) \in A$, the individuals $\overline{a}$ occur in Φ.

[3]Other loosely related work includes results on computational complexity of concept abduction in DL $\mathcal{EL}$ [Bienvenu, 2008] and some proposals concerning other forms of non-monotonic reasoning in DL, such as debugging [Schlobach et al., 2007] or circumscription [Bonatti et al., 2006].

The definition generalizes the one proposed in [Elsenbroich et al., 2006] by allowing multiple assertions as parts of an abductive query, and extends it with the known from the previous chapters constraints of minimality and consistency. At this stage we drop the relevance condition, which, as more problematic, will be treated separately in the subsequent section. A novel aspect in the above characterization with respect to Definition 3 is the introduction of a new category of solutions. As the condition states, solipsistic solutions may involve assertions only about individuals that are explicitly addressed in abductive queries. Surely, it is not reasonable to consider only assertions having this feature, however, the distinction can turn out useful in certain contexts. Finally, we explicitly distinguish between the DL languages that serve for expressing the knowledge base, the query and solutions of an abductive problem. Even though some other possibilities might be interesting from the theoretical viewpoint, in practice, the only feasible ordering of these languages with respect to their expressive power is $\mathcal{L}_Q, \mathcal{L}_S \preceq \mathcal{L}$, as only then a solution and the query can be harmlessly embedded in the knowledge base. In fact, in a similar manner as before, we will be restricting both $\mathcal{L}_Q$ and $\mathcal{L}_S$ to the conjunctive variant of language $\mathcal{L}$.

As a short illustration of an ABox abduction problem consider the following DL ontology and abductive query: $\textsc{Happy}(\mathsf{John})$.

$$\textsc{Optimist} \sqcup (\textsc{Nihilist} \sqcap \exists \mathsf{owns}.\textsc{Dog}) \sqsubseteq \textsc{Happy}$$
$$\textsc{Ignorant} \sqsubseteq \textsc{Optimist}$$
$$\textsc{Nihilist}(\mathsf{John})$$
$$\textsc{Dog}(\mathsf{Snoopy})$$

Even for such a small knowledge base there are several minimal non-trivial solutions to the problem. We can for instance conjecture that $\textsc{Optimist}(\mathsf{John})$ or more specifically that $\textsc{Ignorant}(\mathsf{John})$, which, given the TBox, both entail that $\textsc{Happy}(\mathsf{John})$. Also, since $\textsc{Nihilist}(\mathsf{John})$ is already in the ABox, the query is entailed by assertion $\exists\mathsf{owns}.\textsc{Dog}(\mathsf{John})$. We can further specify that statement by involving a known individual, e.g. $\mathsf{owns}(\mathsf{John},\mathsf{Snoopy})$, or by abducing the existence of a new object: $\mathsf{owns}(\mathsf{John},\mathsf{new\_ind})$, $\textsc{Dog}(\mathsf{new\_ind})$. Clearly, the last two solutions are not solipsistic, because they contain assertions about individuals that are not part of the query. Moreover, the latter is also one that could be called *creative* [Elsenbroich et al., 2006] as it extends the signature of the knowledge base with a new individual name.

There are at least three reasons for which investigating abduction in DL, and ABox abduction in particular, is a worthwhile endeavor. First of all, as emphasized in the introduction of the thesis, there is a high demand in the field of DL and Semantic Web technologies for tools providing non-monotonic reasoning services. Abduction comes here as one of the most general styles of such an inference, which can easily handle a variety of reasoning tasks and application scenarios, both from the user, as well as the ontology engineering perspective. The range of fields that find DL-based techniques useful, or even

indispensable, is already impressive (e.g. e-Science, medical informatics, law and AI, computational linguistics, computer-supported engineering and design) and it is evidently growing. Moreover, every one of them has its own ways of exploiting abductive forms of reasoning, which comprise an essential part of the field's methodology. Hence, the work on abductive DL reasoners fits perfectly in the demands of the broadly understood KR market. Obviously, the design of suitable reasoning tools require a thorough examination of the formal, especially logical and computational foundations of the underlying inference. Such an analysis is precisely the subject of this research.

Further, even if one agrees as to the importance of the research on abduction in DL in general, it might be still questionable why ABox abduction should be addressed as a starting point. Our claim is that ABox abduction is in fact the most fundamental form of abductive reasoning, in the sense that other types of abduction (at least from the classification presented above) can be reduced to ABox abduction problems. The reduction can be roughly characterized as follows:

***Concept abduction*:** given a concept abduction problem $\langle \mathcal{K}, C \rangle$ solve ABox abduction problem $\langle \mathcal{K}, C(a) \rangle$, where a is a new individual name. For any solipsistic solution $H(a)$ to $\langle \mathcal{K}, C(a) \rangle$, H is a solution to $\langle \mathcal{K}, C \rangle$. In case of a conditionalized concept abduction problem, where D is the conditioning concept, prior to solving $\langle \mathcal{K}, C(a) \rangle$ assert $D(a)$, and solve the problem as before.

***TBox abduction*:** given a TBox abduction problem $\langle \mathcal{K}, C \sqsubseteq D \rangle$ solve two concept abduction problems: $\langle \mathcal{K}, \neg C \rangle$ and $\langle \mathcal{K}, D \rangle$. For any two solutions E to $\langle \mathcal{K}, \neg C \rangle$ and F to $\langle \mathcal{K}, D \rangle$, $\neg E \sqsubseteq F$ is a solution to $\langle \mathcal{K}, C \sqsubseteq D \rangle$. Alternatively, solve concept abduction problem $\langle \mathcal{K}, \neg C \sqcup D \rangle$. For any solution E to $\langle \mathcal{K}, \neg C \sqcup D \rangle$, $\top \sqsubseteq E$ is a solution to $\langle \mathcal{K}, C \sqsubseteq D \rangle$.

***Knowledge base abduction*:** given a knowledge base abduction problem $\langle \mathcal{K}, \varphi \rangle$, solve it via combination of TBox and/or ABox abduction problems.

Clearly, all results concerning ABox abduction, can therefore automatically shed light on the formal nature of the other forms of abductive reasoning in DL.

Finally, apart from the aforementioned practical objectives, which remain the central motivation for our work, there is also a strong theoretical rationale for researching abduction in DL, which we only want to signal here. Interestingly, there is a nice overlap of the classification of abduction in DL with the philosophical perspective on abductive reasoning in science. Some authors argue there are two basic models of abductive reasoning as it occurs in the scientific inquiry [Schurz, 2002, Thagard, 1988]. The first one, denoted as *factual abduction*, embraces forms of reasoning from observations to explanatory facts. This model belongs to the, what might be called, normal or low-level methodology of scientific practice, i.e. cases where scientific laws and theories are applied in order to provide explanations of observable phenomena, according to the current

state of knowledge. A special type of factual abduction is *first order existential abduction*, involved when an explanation requires postulating a new entity (e.g. a new unobserved celestial object that explains abberations in the motion of others). The second model represents *law abduction*, with a special case of *second order existential abduction*. This scheme of reasoning comprises the innovative or high-level methods used in science, when new laws, theories or novel concepts are formed, extending the theoretical basis of scientific knowledge.

Under the standard KR interpretation of the layered structure of DL knowledge bases (i.e. TBox represents the common and fixed ontology of a domain, while ABox contains descriptions of the domain objects expressed in terms of that ontology) the two models of abduction in science turn out to tightly correspond to the ABox and TBox abduction. The analogy holds in many dimensions, for instance, in the logical characterization of the two models: one operating on ground instances, the other — on universally quantified FOL formulas. This suggests an interesting way of bridging the gap between the two, supposedly remote fields. Our belief is that Description Logics provide an excellent environment for continuation of the program of *computational philosophy of science*, an unorthodox offspring of the traditional field of philosophy, descending from works of H. Simon [Simon, 1977], and developed and popularized by P. Thagard [Thagard, 1988] and others. The results that can stem from such a coupling are likely to benefit both sides. Computational philosophers of science can finally obtain a standardized, formally delimited framework for pursuing the analysis of scientific methodology, the lack of which was the obvious bottleneck of the discipline. The DL community can in turn gain valuable insights into epistemological foundations and mechanisms of non-monotonic reasoning, or even particular heuristic methods,[4] which are essential for building efficient and practical automated reasoners, and which otherwise, unlike in case of deduction, cannot be justified on purely logical and computational grounds.

4.3 Adaptation issues

We have now introduced all the preliminaries necessary for getting a right grip on the main task — adapting the computational framework for abduction, discussed in Chapter 3, to ABox abduction problems in DL. Clearly, an additional formal machinery will be required to get from the propositional case to much more expressive DL. One strong intuition is to utilize the correspondence between DL and FOL or modal logic. Following that route, we should try lifting the framework to one of these logics (which should not be in principle very difficult) and thus be able to solve ABox abduction problems as special cases of

[4]Notice, that some ideas suggested in Section 3.5 are already borrowed from the literature on philosophy of science. For instance, the strategy for reducing the entropy of a set of solutions by adjudicating between their deductive consequences is actually a straightforward adaptation of the cross-experiment method, evoked in the context of the theory choice problem, and more generally, an application of Popper's hypothetico-deductive model of corroboration of scientific theories [Popper, 1959].

abductive problems expressed in a corresponding formalism. Let us take FOL as the first candidate and further elaborate on the idea.

Encouragingly, both abductive procedures can be easily accommodated to FOL, resulting in two sound and complete calculi for minimal and consistent abduction. Naturally, the termination of the algorithm cannot be guaranteed anymore, but that is fine: as long as DL remains a decidable subset of FOL, and it is the only subset of interest to us, termination can be hopefully regained by employing a mechanism analogous to DL blocking rules [Baader and Sattler, 2001].

In order to show that extension to FOL is possible, it has to be demonstrated that both underlying reasoning techniques are refutation complete for full FOL, and also, some slight refinements to the way of handling solutions have to be introduced. The first prerequisite is granted by respective *lifting theorems*, which are standard arguments, resting on *Herbrand's theorem*, used to show that a proof system that is refutation complete for ground instances of FOL formulas (propositional logic) is also complete for full FOL, provided the instantiation procedure is fair. Fairness means simply that no formula and no instance of it, such that it could be used in the proof, is persistently omitted in the selection process. The proof of the lifting theorem for regular connection tableaux can be found e.g. in [Hähnle, 2001] and for resolution with set-of-support in [Loveland, 1978].

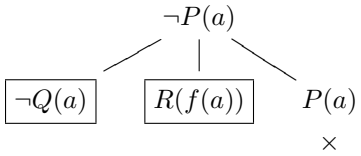
From the procedural perspective the shift to FOL requires, moreover, a practical strategy for dealing with the process of instantiation. The most feasible approach in case of procedures restricted to clause input, such as ours, is to use *Skolemization* of formulas, i.e. transformation to Skolem Normal Form, prior to CNF reduction. A formula in SNF contains only universal quantifiers placed in its front. The transformation runs through reduction to Prenex Normal Form (pushing quantifiers to front and the negation symbol inside) and replacement of existentially bound variables with so-called *Skolem terms* — functional terms that use the universal variables bound before the existential one as their arguments. In fact, one can drop universal quantifiers as well, noting that all *free variables* are implicitly bound. Whenever a clause is introduced to a proof, all its variables should be uniquely renamed to avoid interfering with the ones already present in the proof. Further, unification techniques should be engaged to guide the substitution of free variables.

Covering the unification part in tableaux and resolution is quite straightforward. Since both procedures are based on connections, it is necessary to strengthen this notion first.

Definition 14. Two complementary literals form a *connection* in a proof if they are unified in the proof with a *Most General Unifier* (MGU).

This alone sufficiently restricts the binary resolution rule and beta expansion rule for tableaux. Similar conditions need to be imposed also on the two remaining rules (factoring and branch closure), ensuring that they can be applied only to unifiable literals. In resolution we also have to lift the redundancy elimination criteria. This accomplishes the procedural lifting of both systems to FOL.

The second adaptation step for the abductive framework concerns retrieving correct solutions from the refutation proof of the query. The method typically recommended for that purpose is known as *reverse Skolemization* (cf. [Mayer and Pirri, 1993, Paul, 1993]). As the name suggests, the procedure comes down to reversing the effects of Skolemization (assuming we consider the complement of a Skolemized formula). Every Skolem term has to be replaced by a new universally bound variable, whereas every variable by an existentially bound one. Other constants, which have not been introduced via Skolemization, should be left unchanged. As an example consider a simple abductive problem in FOL: $\langle\{\forall_x(Q(x) \wedge \neg\exists_y R(y) \rightarrow P(x))\}, P(a)\rangle$. After the SNF transformation, the knowledge base contains formula $\neg Q(x) \vee R(f(x)) \vee P(x)$, which is already in CNF. The figure below presents the only possible connection tableau (with the necessary MGU substitutions already applied) for the problem:

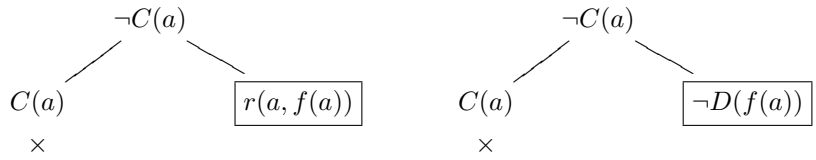


The leaves from the open branches $\Sigma_T = \{\neg Q(a), R(f(a))\}$ should be consecutively negated and reversely Skolemized. As a result we obtain a single solution: $\forall_x(Q(a) \wedge \neg R(x))$, which clearly entails the query.

The two abductive procedures, augmented with the reverse Skolemization mechanism, are sufficient for performing sound and complete abductive reasoning in FOL. Let us then try to solve ABox abduction problem $\langle\{\forall r.D \sqsubseteq C\}, \{C(a)\}\rangle$ using one of the methods and applying the standard translation forth and back. First, we transform the knowledge base in three stages.

1. ST: $\{\forall_x((\forall_y r(x,y) \rightarrow D(y)) \rightarrow C(x))\}$
2. Skolemization: $\{(r(x, f(x)) \wedge \neg D(f(x))) \vee C(x)\}$
3. CNF: $\{r(x, f(x)) \vee C(x), \neg D(f(x)) \vee C(x)\}$

After that, we proceed with the construction of possible FOL tableaux.

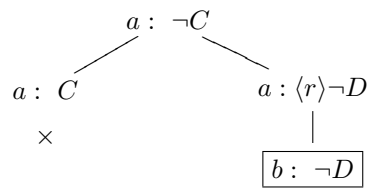


The two FOL solutions are: $\forall_x \neg r(a,x)$ and $\forall_x D(x)$. Finally, we should translate them back to DL, but here is where the troubles begin. The two formulas do not have proper DL counterparts. On a more careful look, one can

notice that the expected DL solution $\forall r.D(a)$, corresponding to FOL formula $\forall_x(\neg r(a,x) \vee D(x))$, has been broken into two parts, each of which has been selected as an independent solution, inexpressible in the target language. The reason for this somewhat surprising effect is that the computation method we use is essentially incapable of finding solutions in FOL disjunctive form, such as involving DL's universal restrictions.[5] We could in principle fill in the lacking parts of the FOL solutions, while preserving the semantic equivalence to the original ones, and assert $\forall r.\bot(a)$ and $\forall u.D(a)$[6] instead, but nonetheless, we would still miss the point. The solution we want to express first of all is neither "a does not have any r-successors" nor "all individuals in the domain are D", but "all r-successors of a are D".

The role component of DL universal restrictions plays in fact a very specific function, whose meaning can be fully grasped only on the grounds of relational semantics. Informally speaking, it is not another predicate asserted about some individuals in the domain, but a metalinguistic pointer that narrows down the interpretation of the predicate only to the subset of objects — exactly those that are linked to a particular individual via the given relation. It turns out that reverse Skolemization is too crude a tool for the logics we are dealing with. Observe, that solutions returned by that mechanism can only provide statements about a named individual (e.g. $P(a)$), about an existing but unknown individual (e.g. $\exists_x P(x)$) or about all individuals in the domain (e.g. $\forall_x P(x)$), but clearly not about their subsets. Therefore, the whole range of DL solutions, which contain nested quantification restrictions, is inaccessible for the general FOL-based abduction.

Since preservation of the underlying relational structure is crucial for solving abductive problems in DL, let us consider an alternative approach, resting on the correspondence of DL to multimodal logic K_n. In [Mayer and Pirri, 1995] a tableau procedure for abduction in modal logic K has been developed and proven sound and complete. The substitute to reverse Skolemization, proposed in that paper, is simply bookkeeping of the sequence of expansion steps for boxed formulas on branches. Solutions are to be reconstructed by folding back the boxes from branches in the reverse order and with reversed modalities. Let us provisionally adopt this idea, leaving aside several incompatibility issues for the time being, and return to the example. Problem $\langle\{\forall r.D \sqsubseteq C\}, \{C(a)\}\rangle$ can be now translated through π^K and CNF to $\langle\{\langle r\rangle\neg D \vee C\}, \{a : C\}\rangle$. This leads to the following tableau tree:



[5] Note, that resolution-based abduction would result in the same outcome.

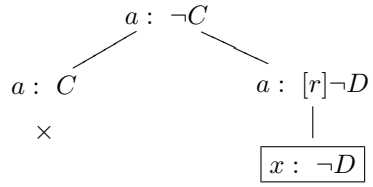
[6] Role u denotes here the so-called *universal role* defined as $u = \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, which is a default component of DL $\mathcal{R}$ languages.

In order to retrieve the solution from the tableau we first negate the leaf, obtaining D, and then move up its branch collecting expanded boxes. On encountering $\langle r \rangle$, we modify the solution to $[r]D$ and finally reach the root labeled with a, which results in the final outcome: $a : [r]D$. The assertion neatly translates back to $\forall r.D(a)$.

Apparently, in its core, the modal approach is much more constructive and provides a better starting point for the DL abduction framework. Nevertheless, it gives rise to many other practical problems. Basically, standard modal proof systems are not well suited for connection-driven theorem proving, although it is in principle possible [Waaler, 2001]. Typically, a tableau modal proof progresses alongside the branches of the underlying tree-shaped model, thus likely loosing connections. Naturally, we would like to maintain the property of connectedness in order to keep the procedure goal-oriented. For that purpose it would be necessary to thoroughly revise the standard methods, to account at least for the following aspects:

1. specifying the clause form of the input, required for a convenient access to connections,
2. introducing limited Skolemization, enabling a reuse of the same $\Diamond$-formulas in different parts of the proof,
3. allowing the use of free variables for expansion of $\Box$-formulas.
4. elaborating a way of handling multiple modalities, also keeping in mind expressive extensions including role hierarchies, inverse roles, etc.

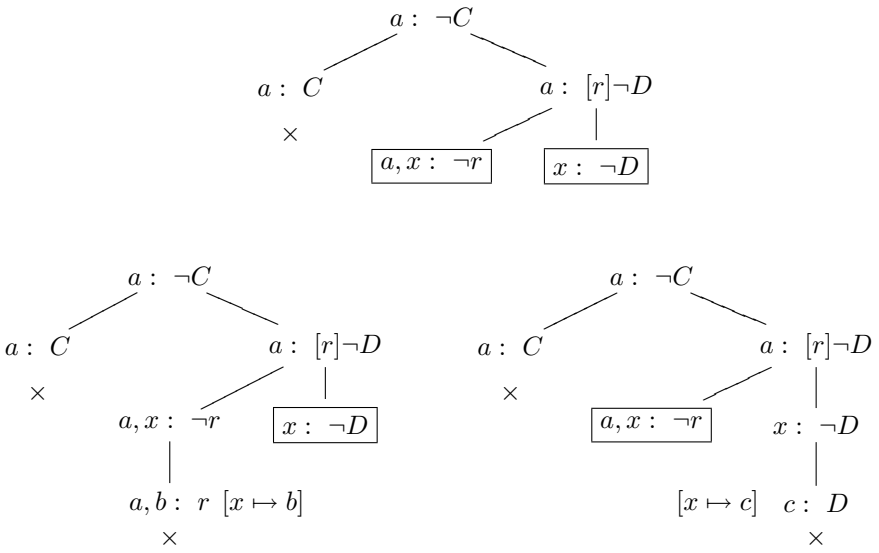
Assuming that a suitable method could be devised with some effort, there is still another serious problem. Despite the established correspondence, the expressiveness of DL ABoxes goes well beyond the expressiveness of a multimodal logic K_n. Observe, that assertions admissible as solutions in ABox abduction, are not only, or not really modal formulas, but rather pieces of Kripke models: formulas true in particular worlds (sometimes newly postulated worlds) and fragmentary definitions of accessibility relations. Especially, the latter issue is problematic as it requires roles to be explicitly represented in proofs. To illustrate this situation consider ABox $\mathcal{A} = \{r(a,b), D(c)\}$, TBox $\mathcal{T} = \{\exists r.D \sqsubseteq C\}$ and problem $\langle \mathcal{K}, \{C(a)\} \rangle$. Having translated the TBox to $\mathcal{T} = \{[r]\neg D \vee C\}$ the following modal-like connected tableau, with a free variable x, can be constructed:



Let us compare three of the possible solutions:

1. $r(a,d), D(d)$
2. $D(b)$
3. $r(a,c)$

The first one introduces a new individual d as an r-successor of a, whereas the other two use existing individuals, and merely supplement their descriptions with the missing attributes required for being an r-successor of a of type D. Clearly, some of the information expressed in the solutions is implicit in the tableau tree. It is not clear from the representation in which cases the role assertion has to be added to a solution and in which not. A more faithful account of the inference should be based on the following tableaux:



Explicit representation of roles is even more desirable once DLs $\mathcal{H}$ or $\mathcal{I}$ are considered, which involve reasoning over role constructs, equally to reasoning over concepts.

The last comments of this section regard the constraints of minimality and relevance. Notice, that the method of $\subseteq$-ordering comparison, used in the propositional case for assessing their satisfaction, is clearly insufficient for DLs. As long as the assessment takes place in the FOL translation over Skolemized formulas it is still possible to lift the criterion as follows:

Proposition 5. *For any two Skolemized FOL conjunctive formulas α and β, such that α and β are satisfiable, it holds that $\alpha \vDash \beta$ iff there exists an MGU σ such that $Cn_\beta \subseteq \sigma(Cn_\alpha)$.*

However, the proposition is not very helpful when we allow more complex syntactic forms, such as $\exists r.C(a)$ in DL ABox. Observe that instead of asserting

$\exists r.C(a)$ one can equally well abduce the expanded variant of the statement: $r(a,d), C(d)$, where d is a new individual name or the name of an existing individual. Similarly, if $\exists r.C(a)$ is the query of an abductive problem, one can trivially solve it with $r(a,d), C(d)$. In both cases we lack a practical procedure of comparing the solutions and inferring that in fact $\exists r.C(a)$ is minimal, and $r(a,d), C(d)$ is moreover not relevant, since for any d, it follows that $r(a,d), C(d) \models \exists r.C(a)$ but $\exists r.C(a) \not\models r(a,d), C(d)$. It is easy to find out that the following statements are also true.

1. $\exists r.(C \sqcap D)(a)$ is non-minimal w.r.t. $(\exists r.C \sqcap \exists r.D)(a)$.
2. $r(d,a), \forall r.C(d)$ is non-minimal w.r.t. $C(a)$.

In this work we will not propose an efficient procedural approach to verifying minimality, leaving the issue on the level of semantic definition.

Relevance is a source of further concerns. Expressiveness of DL offers more syntactic possibilities of constructing irrelevant solutions than propositional logic. A very characteristic one is involved in DL tautology $\exists r^-.(\forall r.C) \sqsubseteq C$, for any r and C. Hence, for query $C(a)$, $\exists r^-.(\forall r.C)(a)$ is not a relevant solution. Even in languages without inverse roles it is possible to mimic the structure of the underlying model by asserting $r(d,a), \forall r.C(d)$. Note, that unless we explicitly insert the tautology into the knowledge base, this form of solutions will not be found through neither of the abductive procedures. Naturally, we would like to eradicate irrelevance. Alas, the tautology can serve for constructing a "partially" irrelevant solution, similarly to the propositional case. For instance problem $\langle \{B \sqsubseteq D\}, \{(C \sqcap D)(a)\} \rangle$ is solved by $(\exists r^-.(\forall r.C) \sqcap B)(a)$, though the first part of the intersection is a somewhat ad hoc implicant of $C(a)$. To be exact, the solution is still non-minimal with respect to $(C \sqcap B)(a)$, and so it can be safely dropped. The following example, however, in which we use one more tautology $\exists r.\top \sqcap \forall r.C \sqsubseteq \exists r.C$ and TBox axiom $D \sqsubseteq \exists r^-.\top$, proves that tautologies can lead to minimal solutions, that cannot be obtained in any other way. Let the problem be specified as $\langle \{D \sqsubseteq \exists r^-.\top\}, \{C(a)\} \rangle$. Like previously $C(a)$ is entailed by $\exists r^-.(\forall r.C)(a)$, and this in turn by $(\exists r^-.\top \sqcap \forall r^-.(\forall r.C))(a)$. Finally we replace $\exists r^-.\top$ with D, according to the TBox axiom, and obtain solution $(D \sqcap \forall r^-.(\forall r.C))(a)$ which is both relevant and minimal.

Seemingly, we face here a hard design choice, with at least three options:

1. Identify all DL tautologies that can bring about similar effects and add them in form of axiom schemes to every knowledge base while solving a problem.
2. Relax the requirement for logical completeness of the procedures.
3. Strengthen the notion of relevance to leave out this kind of constructs from possible solution spaces.

We shall leave the choice unresolved, though our strong preference is directed towards the third option. Arguably, the first solution is very unpractical, as it

would certainly lead to overflow of redundancy in proofs, whereas the second one actually avoids the problem. Since some sort of abductive solutions is about to be discarded, the selection should be backed with a strict formal justification and criterion, rather than be left as a loose end of the procedure.

In Section 2.2, we have proposed a definition of *strong relevance*, which is aimed at discarding also partially irrelevant solutions, i.e. those which by their own convey some portion of the information content of the query. We will show that the notion is sufficient to eliminate (at least some of) the problematic solutions. Recall, that α is strongly relevant solution to $\langle \mathcal{K}, \varphi \rangle$ iff for every γ, δ (in conjunctive form) such that $\varphi \models \gamma$ and $\varphi \models \delta$ if $\{\alpha\} \cup \{\gamma\} \models \delta$ then $\gamma \models \delta$. Consider the query $C(a)$ and its two consequences, namely: $C(a)$ and $(\forall r^{-}.(\exists r.\{a\}))(a)$. We have found that $(D \sqcap \forall r^{-}.(\forall r.C))(a)$ is an undesired solution to $C(a)$. But notice that $\{(D \sqcap \forall r^{-}.(\forall r.C))(a)\} \cup \{(\forall r^{-}.(\exists r.\{a\}))(a)\} \models C(a)$, but it is not true that $(\forall r^{-}.(\exists r.\{a\}))(a) \models C(a)$. Therefore, the solution is not strongly relevant. As promising as it may seem, the strong relevance condition is unfortunately too strong, and eliminates also some interesting solutions. For instance $\exists r.B(a)$ is not a strongly relevant solution to $\langle \{B \sqsubseteq C\}, \{\exists r.C(a)\} \rangle$, as the assertion of existence of an r-successor of a is already contained in it.

In the following we will not address issues related to minimality and relevance anymore, assuming that an appropriate adjustment of their semantic formulation as well as obtaining an effective procedure for checking their satisfaction in the context of ABox abduction problems are possible.

Chapter 5

Goal-oriented ABox Abduction

In this chapter we present a goal-oriented procedure for computing solutions to ABox abduction problems in Description Logic $\mathcal{ALC}$. We take into account the insights elaborated in the previous chapters and propose a hybrid method, which accommodates essential features of FOL and modal-style proof systems. In the following sections we discuss consecutive stages of solving ABox abduction problems, including syntactic transformation of knowledge bases, finding solutions and checking their consistency. We also approach the problem of blocking rules for abductive reasoning, necessary to guarantee termination of the procedure for ontologies with cyclic TBoxes. In the last section we suggest some general ideas on how to broaden the framework in order to cover certain expressive extensions of $\mathcal{ALC}$.

5.1 Transformation

A complete transformation of a DL knowledge base into the computation ready form comprises NNF transformation of TBox axioms and concept descriptions in ABox assertions, followed by their CNF reduction, during which we also unfold nested quantification restrictions using special non-DL predicates. Finally all the clauses are Skolemized in a way that allows for preservation of satisfiability.

The idea behind our approach is to obtain a form resembling Skolemized FOL clauses, which is necessary for constructing connection-based proofs, but at the same time to leave the modal character of DL quantification restrictions unaffected, so as to be able to reconstruct relational structure underlying an abductive proof.

We start by extending the signature of the language with a set $\mathbb{F}_{sko} = \{f_1, f_2, \ldots\}$ of Skolem functions and a set $\mathbb{P} = \{P_1, P_2, \ldots\}$ containing non-DL predicates, possibly of different arity. As usual, a function of arity 0 is treated as a constant, whereas a predicate of arity 0 as a proposition. We

also assume there is an infinite set $\mathrm{Var} = \{x_1, x_2, \ldots\}$ of variables. We write $\overline{x} = x_1, \ldots, x_n$ to denote a sequence of variables, and use $\cdot^\star$ to mark introduction of new symbols: $x^\star$ a new variable, $f^\star$ a new function and $P^\star$ a new non-DL predicate.

Below we present an outline of the three stages of transformation in $\mathcal{ALC}$, marked by $\tau_\neg$, $\tau_\sqcap$ and $\tau_x^{\overline{y}}$, for NNF, CNF and Skolemization, respectively. The layering of the process is rather schematic, as in practice it should be much more efficient to interleave steps of transformation steps belonging to different stages, especially those of CNF and Skolemization.

$$
\begin{array}{rcl}
\tau_\neg(C \equiv D) & = & \tau_\neg((C \sqsubseteq D) \sqcap (D \sqsubseteq C)) \\
\tau_\neg(C \sqsubseteq D) & = & \tau_\neg(\neg C) \sqcup \tau_\neg(D) \\
\tau_\neg(\neg\neg C) & = & \tau_\neg(C) \\
\tau_\neg(\neg A) & = & \neg A \\
\tau_\neg(A) & = & A \\
\tau_\neg(\neg\bot) & = & \neg\bot \\
\tau_\neg(\bot) & = & \bot \\
\tau_\neg(\top) & = & \neg\bot \\
\tau_\neg(\neg\top) & = & \bot \\
\tau_\neg(\neg(C \sqcup D)) & = & \tau_\neg(\neg C) \sqcap \tau_\neg(\neg D) \\
\tau_\neg(\neg(C \sqcap D)) & = & \tau_\neg(\neg C) \sqcup \tau_\neg(\neg D) \\
\tau_\neg(\neg\forall r.C) & = & \exists r.\tau_\neg(\neg C) \\
\tau_\neg(\neg\exists r.C) & = & \forall r.\tau_\neg(\neg C) \\
\tau_\neg(C \sqcup D) & = & \tau_\neg(C) \sqcup \tau_\neg(D) \\
\tau_\neg(C \sqcap D) & = & \tau_\neg(C) \sqcap \tau_\neg(D) \\
\tau_\neg(\forall r.C) & = & \forall r.\tau_\neg(C) \\
\tau_\neg(\exists r.C) & = & \exists r.\tau_\neg(C)
\end{array}
$$

Table 5.1: Negation Normal Form transformation.

NNF (see Table 5.1) is standard and does not require any comments. Similarly as in the propositional case (see Section 3.4), having a formula translated into NNF one can easily answer whether it is relevant for a given part of a proof. If such a strategy is employed, the remainder of the transformation can be deferred until a particular connection is requested.

$$
\begin{array}{rcl}
\tau_\sqcap(A) & = & A \\
\tau_\sqcap(\neg A) & = & \neg\tau_\sqcap(A) \\
\tau_\sqcap(C \sqcap D) & = & \tau_\sqcap(C), \tau_\sqcap(D) \\
\tau_\sqcap((C \sqcap D) \sqcup E) & = & \tau_\sqcap(C \sqcup E), \tau_\sqcap(D \sqcup E) \\
\tau_\sqcap(\forall r.C) & = & (\forall r.P^\star), \tau_\sqcap(\neg P^\star \sqcup C) \\
\tau_\sqcap(\exists r.C) & = & (\exists r.P^\star), \tau_\sqcap(\neg P^\star \sqcup C)
\end{array}
$$

Table 5.2: Conjunctive Normal Form transformation.

CNF transformation (see Table 5.2) is also straightforward, excluding, per-

haps, reduction of nested quantification restrictions. In this part we adopt the idea from [Tammet, 1992] of replacing the qualifying concept description of the restriction with a new predicate symbol, which is linked to the description via proper GCI. For instance:

$$\begin{array}{r|l} \forall r.(A \sqcup B) & \forall r.P_1 \\ & P_1 \sqsubseteq A \sqcup B \end{array}$$

$$\begin{array}{r|l} \forall r.(A \sqcap \exists p.(B \sqcup \neg C)) & \forall r.P_2 \\ & P_2 \sqsubseteq A \sqcap \exists p.P_3 \\ & P_3 \sqsubseteq B \sqcup \neg C \end{array}$$

Using this, or a similar technique of dealing with nested restrictions is necessary for facilitating a connection driven search, as only in this form there is a direct access to all literals at any position of the formula.

CNF transformation results in a set of concept unions. With every union $C = C_1 \sqcup \ldots \sqcup C_n$ we associate a set of its disjuncts $Cl_C = \{C_1, \ldots, C_n\}$. Moreover, we distinguish top clauses, i.e. those which do not contain literal $\neg P_i$ for any $P_i \in \mathbb{P}$, marking them as $Cl^\dagger$, and all the remaining, signed as $Cl^\ddagger$. For every TBox axiom φ and for every ABox concept assertion $\varphi(a)$ we first compute the set of clauses $S_\varphi = \{\tau_\sqcap(\tau_\neg(\varphi))\}$, and then proceed with Skolemization, defined recursively in Table 5.3. The process is initiated as follows:

- for every TBox axiom φ and every clause $Cl^\dagger \in S_\varphi$ compute $\tau^z_z(C)$, where $z = x^\star$,
- for every ABox axiom $\varphi(a)$ and every clause $Cl^\dagger \in S_\varphi$ compute $\tau_a(C)$,
- for every ABox axiom $r(a, b)$, leave it unchanged.

$\tau^{\overline{x}}_x(Cl)$	$=$	$\{\tau^{\overline{x}}_x(C) \mid C \in Cl\}$
$\tau^{\overline{x}}_x(A)$	$=$	$A(x)$
$\tau^{\overline{x}}_x(\neg A)$	$=$	$\neg\tau^{\overline{x}}_x(A)$
$\tau^{\overline{x}}_x(\neg P_i)$	$=$	$\neg P_i(\overline{x}, x)$
$\tau^{\overline{x}}_x(\forall r.P_i)$	$=$	$[r(x,z)]P_i(\overline{x}, z)$, where $z = x^\star$ is new in $\overline{x}$
	$\Rightarrow$ for every $Cl^\ddagger$ such that $\neg P_i \in Cl^\ddagger$:	
		$\tau^{\overline{x}}_x(Cl) = \tau^{\overline{x}}_z(\{\neg P_i\}) \cup \tau^{\overline{x},z}_z(Cl \setminus \{\neg P_i\})$
$\tau^{\overline{x}}_x(\exists r.P_i)$	$=$	$\langle r(x, h(\overline{x}))\rangle P_i(\overline{x})$, where $h = f^\star$
	$\Rightarrow$ for every $Cl^\ddagger$ such that $\neg P_i \in Cl^\ddagger$:	
		$\tau^{\overline{x}}_x(Cl) = \tau^{\overline{x}}(\{\neg P_i\}) \cup \tau^{\overline{x}}_{h(\overline{x})}(Cl \setminus \{\neg P_i\})$

Table 5.3: Skolemization.

Observe, that in order to decompose nested formulas in such a manner that satisfiability is preserved and any ambiguity in proofs is avoided, the non-DL predicates have to be used for carrying over all universally bound variables into

separated subformulas, so that Skolem functions, which might possibly occur on the deeper levels of nesting, can have appropriate arguments. Since non-DL predicates are unique for a given restriction, it is guaranteed that the connection can be established in the right place only and that all variables originally shared between a sub– and the superformula will by again fine-tuned by unification.

Although formulas are Skolemized in a typical FOL style, we translate the core structure of the two quantification restrictions into corresponding modal operators: $\forall r.C$ changes into $[r(t_1, t_2)]P(\overline{x})$ and $\exists r.C$ into $\langle r(t_1, t_2)\rangle P(\overline{x})$. The rationale for this apparent incoherency, as explained in the previous chapter, is to make expansion steps for boxed formulas explicit, and thus enable keeping their record, which is necessary for reconstructing abductive solutions in the way respecting the relational semantics underlying DL.

The following table provides an example of a full transformation of three sample DL axioms, according to the scheme $\tau_x^{\overline{y}} \circ \tau_\sqcap \circ \tau_\neg$, from here on τ for short.

$r(a, b)$	$\{r(a, b)\}$
$(C \sqcap \forall r.(\exists p.B))(a)$	$\{C(a)\}$ $\{[r(a, x_1)]P_1(x_1)\}$ $\{\neg P_1(x_1), \langle p(x_1, f_1(x_1))\rangle P_2(x_1)\}$ $\{\neg P_2(x_1), B(f_1(x_1))\}$
$A \equiv D \sqcup \exists p.E$	$\{\neg A(x_1), D(x_1), \langle p(x_1, f_2(x_1))\rangle P_3(x_1)\}$ $\{\neg P_3(x_1), E(f_2(x_1))\}$ $\{\neg D(x_1), A(x_1)\}$ $\{[p(x_1, x_2)]P_4(x_1, x_2), A(x_1)\}$ $\{\neg P_4(x_1, x_2), \neg E(x_2)\}$

Evidently, the set of clauses resulting from full transformation of a knowledge base can be very verbose. However, optimizing the procedure[1] and rendering it connection-driven should significantly diminish the effect.

5.2 Plain abduction

The procedure for finding solutions to ABox abduction problems in $\mathcal{ALC}$ is based on the same techniques as in the propositional case, lifted to full FOL, and augmented with special rules for handling boxed literals. The construction of a proof is accompanied by bookkeeping of the relational structure revealed by expansion of the boxed literals, i.e. original quantification restrictions, occurring in the formulas used in the proof. This structure is encoded in the abductive

[1] Some obvious optimizations include e.g. 1) decomposing nested restrictions from formulas in NNF before any distribution rules are applied; 2) carrying over variables inside the embedded clauses only down to the level where last existential restriction occurs; 3) skipping decomposition of restrictions whose qualifying concept is a literal.

graph associated with the proof. Further, we suitably refine the approach to reconstructing solutions from incomplete proofs. The method is slightly more complex than before. Basically, not every proof leads to an expressible solution. We therefore identify and formulate two conditions for this to be the case: one regarding the set of literals obtained in a proof (DL-admissibility), the other concerning the associated graph ($\mathcal{ALC}$-admissibility). Finally, we present an inference procedure $\vdash_{\text{ABox}}$ that, given the conditions are satisfied, maps the proof into a corresponding set of ABox assertions solving the abductive problem.

The syntactic form of solutions and queries will be restricted to DL language $\mathcal{ALE}$, which is the conjunctive variant of $\mathcal{ALC}$. In $\mathcal{ALE}$, the union constructor is prohibited and use of negation is limited only to atomic concepts (see below). Otherwise, the language does not introduce new expressive means with respect to $\mathcal{ALC}$.

$$\top \mid \bot \mid A \mid \neg A \mid C \sqcap D \mid \forall r.C \mid \exists r.C$$

Table 5.4: $\mathcal{ALE}$ concept constructors.

Let $\langle \mathcal{K}, \Phi(\overline{a}) \rangle$ be an ABox abduction problem. The search for solutions is initiated by negating every assertion in Φ and processing the resulting set via τ transformation. Since assertions in Φ are implicitly connected with the intersection operator, we take clause $Cl_{\text{Init}} = \bigcup \tau(\overline{\Phi})^{\dagger}$ (the union of all top clauses from transformation of $\overline{\Phi}$) as the initial clause of the proof, i.e. the first clause to be expanded on a tableau tree or the clause forming the initial set-of-support. The remaining part of the clauses should be added to the knowledge base.

Tables 5.5 and 5.6 present the inference rules applicable in regular connection tableau and resolution with set-of-support proofs in $\mathcal{ALC}$.

For the resolution method we also adjust the redundancy elimination criteria as follows [Bachmair and Ganzinger, 2001]:

Subsumption At any stage S_i of the run of resolution, for every two clauses $Cl_1, Cl_2 \in S_i$, if there exists a substitution σ such that $\sigma(Cl_1) \subseteq Cl_2$ then Cl_2 can be removed from S_i.

Tautology deletion At any stage S_i of the run of resolution, for every clause $Cl \in S_i$, if there exists a substitution σ such that $\sigma(Cl)$ is a tautology then Cl can be removed from S_i.

We assume that every clause added to a tableau is made variable disjoint with it, and similarly, before creating a resolvent, the two clauses (excluding the connecting literals) are also made variable disjoint. Apart from the standard rules, strengthened with the unification requirement, both systems are equipped with new rules for expanding boxed literals. Since the role component and the non-DL predicate comprising a boxed literal are already appropriately

$$\frac{Cl = \{L_1, \ldots, L_n\} \in \mathcal{K} \cup \mathcal{K}^{\nu,\pi} \quad \vdots \quad \overline{L}}{L_1 \mid \ldots \mid L_n} \qquad \frac{L_1 \quad \vdots \quad \overline{L_2}}{\bot} \qquad \frac{\vdots \quad \bot \quad \vdots}{\bot}$$

$\beta - \textbf{rule}$ **Branch closure**

iff there exists an MGU σ of L and L_i for some $i \in (1, \ldots, n)$ (or L_1 and L_2), and σ is applied to the whole tableau

$$\frac{[r]P}{\neg r \mid P} \qquad \frac{\langle r \rangle P}{r} \qquad \frac{\langle r \rangle P}{P}$$

$\nu - \textbf{rule}$ $\pi' - \textbf{rule}$ $\pi'' - \textbf{rule}$

Table 5.5: DL tableau rules.

Skolemized, the application of rules comes down to breaking boxes into their corresponding FOL representation: $[r]P$ into $\neg r \vee P$ and $\langle r \rangle P$ into $r \wedge P$. Note, that the π rule for tableau is nondeterministic, as it allows for extraction of either the role or the predicate. For that reason it is necessary to relax the regularity constraint for tableaux, so that the same $\Diamond$-literal could occur twice on the same branch.

Construction of proofs is in principle connection-driven. In certain situations, however, in order to establish a connection between two clauses, one of them has to be expanded with ν or π rule prior to being connected to the proper proof. Consider for instance $Cl_1 = \{A(a), \langle r(a, f(a)\rangle P(f(a))\} \in \mathcal{K}$ and $Cl_2 = \{\neg P(x), B(f(x))\} \in T$, where T is either a tableau or the set-of-support of a run of resolution. Clearly, Cl_1 can be expanded to a relevant clause $\pi''(Cl_1) = \{A(a), P(f(a))\}$, which can form a connection with Cl_2 in T. Although ν and π rules cannot be in general applied to clauses from outside of T, we allow, as a matter of exception, such connections to be established. The set of all clauses obtained by ν, π' or π'' expansion of the clauses from $\mathcal{K}$, which unless they were expanded could not be connected to the proof at a given stage, will be denoted as $\mathcal{K}^{\nu,\pi}$. In practice there is no need to compute set $\mathcal{K}^{\nu,\pi}$, but include boxed literals in the search for connections, knowing moreover, that the situation can occur only when trying to find a connection for a role literal or for a non-DL predicate.

With the set of leaves on open branches of a tableau tree and every clause in the set-of-support in a run of resolution we associate a directed graph $\mathcal{G} = (V, E)$, whose vertices are terms (variables, individual names, Skolem terms) and edges are labeled with role names. A graph represents the family of modal frames underpinning the proof of a particular clause, which serves for appropriate re-

$$\frac{Cl_1 \cup \{L_1\} \in S \cup \mathcal{K}^{\nu,\pi} \quad Cl_2 \cup \{\overline{L_2}\} \in T}{\sigma(Cl_1 \cup Cl_2) \in T} \qquad \frac{Cl \cup \{L_1, L_2\} \in T}{\sigma(Cl \cup \{L_1\}) \in T}$$

Binary resolution **Factoring**

where σ is an MGU of L_1 and L_2

$$\frac{Cl \cup \{[r]P\} \in T}{Cl \cup \{\neg r, P\} \in T} \qquad \frac{Cl \cup \{\langle r \rangle P\} \in T}{Cl \cup \{r\}, Cl \cup \{P\} \in T}$$

$\nu -$ **rule** $\pi -$ **rule**

Table 5.6: DL resolution rules.

construction of the solution correlated with the clause.

Graphs are initiated by posing the root graph: $V = \{\overline{a} \mid \Phi(\overline{a})\}$ and $E = \{r(t_1, t_2) \mid r(t_1, t_2) \in \Phi\}$ for the first clause in a proof, which is then copied and extended along the search for solutions, recording all ν and π expansion steps made in the proof. On every application of ν and π rule to $[r(t_1, t_2)]P(\overline{x})$ or $\langle r(t_1, t_2)\rangle P(\overline{x})$, either inside of the proof, or while connecting a clause from $\mathcal{K}^{\nu,\pi}$, the resulting graph is extended with vertex t_2 and edge $r(t_1, t_2)$. Graphs are also supposed to be substitution sensitive, i.e. whenever a variable is substituted with a term in the proof, the relevant graphs should account for this operation as well. On reconstructing the solution every variable and individual name in a graph is treated as an individual object, whereas every Skolem term is interpreted as a set of objects bound by certain $\forall$ operator. This way it is possible to support reverse Skolemization, which respects the underlying relational structure.

The expressive power of $\mathcal{ALC}$ delimits the scope of graphs that can be faithfully incorporated into an expressible solution. Basically, there are three conditions for a graph's structure that have to be guaranteed:

Definition 15 ($\mathcal{ALC}$–admissible graph)**.** Graph $\mathcal{G} = (V, E)$ associated with a clause Cl in a proof T of an abductive procedure is $\mathcal{ALC}$*–admissible* iff all the following requirements are satisfied:

- for every Skolem term $t \in V$ there is a unique $t' \in V$ and r, such that $r(t', t) \in E$,
- for every Skolem term $t \in V$, t can be only succeeded by a tree-like structure in $\mathcal{G}$,
- no names N_I occur as vertices in those trees.

The rationale behind the restrictions should be obvious. DL formulas, like their modal counterparts, have a tree-model property. Since Skolem terms can be expressed only via a formula starting with a universal restriction, all their

successors have to be ordered in a way that can be captured by a complex formula — clearly a tree. Moreover, every formula needs to have a single root, hence the single predecessor requirement. Finally, as long as we do not have nominals on our disposal, all individuals belonging to a formula's model are left anonymous, hence no names from N_I can be of use there.

On the contrary to Skolem terms, relationships between named individuals and variables can take structures of arbitrary shapes, which can all be expressed via ABox assertions of the form $r(t_1, t_2)$. Some of them might inevitably lead to non-minimal solutions, e.g. such as involving multiple edges between two variables, but nevertheless they are all expressible. Figure 5.1 presents an example of an $\mathcal{ALC}$–admissible graph that could be associated with a solution to some query $\varphi(a)$.

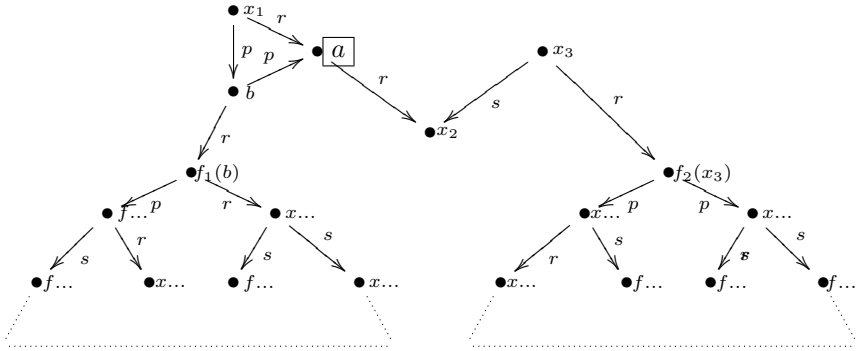


Figure 5.1: An $\mathcal{ALC}$–admissible graph for a solution to $\varphi(a)$.

Introduction of graphs evokes another very interesting notion of minimality, which could be of high relevance in the context of DL knowledge bases. *Domain minimality* enforces a preference for solutions that assume the smallest domain, or in other words, that abduce existence of the minimum number of new individuals. The notion, occurring sometimes in discussions on *circumscription* and *model generation* (cf. [Lorenz, 2004], [Garcia et al., 2007]), apart from its obvious Occamist flavor, has also a vital computational justification. Basically, it allows to minimize the increase of complexity of a knowledge base (known to be strongly dependent on the size of domain) as a result of solving an abduction problem. On the other hand it is clearly traded off against the standard notion of minimality, to which we are committed here. Domain minimality would require, for instance, that any two variables that can be consistently unified in the proof, were so, whereas the standard notion considers such a unification as an attempt to convey more information than is actually necessary for the query to be entailed. Tracing and balancing this trade-off is certainly a valid issue for abduction in DL, although we will not address it here.

Apart from discarding clauses associated with not admissible graphs, we will also place restrictions on the set of literals comprising a clause from which an

ABox solution can be constructed.

Definition 16 (DL-admissible clause)**.** Let Cl be a clause of literals obtained in an abductive procedure for an ABox abduction problem. Cl is DL–*admissible* iff Cl does not contain any of the following literals:

- $P_i(\overline{x})$ or $\neg P_i(\overline{x})$ for any $P_i \in \mathbb{P}$,
- $[r(t_1, t_2)]P_i(\overline{x})$ or $\langle r(t_1, t_2)\rangle P_i(\overline{x})$ for any $P_i \in \mathbb{P}$ and any r,
- $r(t_1, t_2)$ for any r,
- $\neg r(t_1, f_i(t_2))$ for any $f_i \in \mathbb{F}$ and any r.

The first two conditions are straightforward: it is impossible to include non-DL predicates into DL assertions. The third restriction is somewhat conventional, but it also plays an important role in the proof of completeness of the abductive procedures. Notice, that $r(t_1, t_2)$ might have occurred in a clause only due to application of π'' rule and so, after reconstructing the solution, it has to become a part of some universal restriction. But information about $r(t_1, t_2)$ is already encoded in the graph, and what is rather unknown is the qualifying concept of the restriction. Hence, one can replace the π'' expansion step with π', to have all necessary information available. Finally, the last condition discards clauses that contain a construct inexpressible in DL, as the second term in the role assertion would have to be universally quantified.

We can now describe the procedure of retrieving an $\mathcal{ALC}$ solution and thus define a new notion of a $\vdash_{\text{ABox}}$ plain solution to an ABox abduction problem. Roughly, the idea is to fold up all tree-like structures into complex concept assertions, possibly leaving out some redundant branches, and add remaining atomic concept and role assertions between existing and/or new individuals.

Definition 17 ($\vdash_{\text{ABox}}$-abduction)**.** Let $\langle K, \Phi\rangle$ be an ABox abduction problem, Cl a DL-admissible clause of literals obtained in an abductive procedure for $\tau(\mathcal{K} \cup \overline{\Phi})$ initiated by clause $\bigcup \tau(\overline{\Phi})^{\dagger}$, and $\mathcal{G} = (V, E)$ the graph associated with Cl, which is $\mathcal{ALC}$-admissible. The set of assertions A, generated according to the following steps is a *plain solution* to $\langle K, \Phi\rangle$:

1. $A := \emptyset$
2. For every concept assertion $C(t) \in Cl$, add $\overline{C}(t)$ to A and remove $C(t)$ from Cl.
3. For every term $t \in V$, such that it does not have a successor in $\mathcal{G}$, i.e. there is no $t' \in V$ and r such that $r(t, t') \in E$ begin:
   (a) if t occurs in A then replace all concept assertions about t in A with $(\bigsqcap\{C \mid C(t) \in A\})(t)$;
   (b) if t is a Skolem term then find r and t' such that $r(t', t) \in E$ (by Def. 15 there has to be a unique one). If there is $C(t)$ in A then add $(\forall r.C)(t')$ to A and remove $C(t)$. Remove $r(t', t)$ from E;

(c) if t is a variable term and there is a unique r and t' such that $r(t', t) \in E$ then: if there is $C(t) \in A$ add $(\exists r.C)(t')$ to A and remove $C(t)$; else add $(\exists r.\top)(t')$. Remove $r(t', t)$ from E and $\neg r(t', t)$ from Cl;

4. For every $\neg r(t', t) \in Cl$ that is left add $r(t', t)$ to A. Replace all variable terms occurring in A with new individual names in $\mathcal{K}$.

Provided the requirements for admissibility of the clause and its graph are satisfied, the procedure returns a proper set of ABox assertions A and an empty set Cl. Let us comment on the consecutive steps in more detail. First note, that Cl can only contain (positive and negative) atomic concept assertions, and moreover literals of the form $\neg r(t_1, t_2)$, where t_2 is either a variable or an individual name. In step 2 all concept assertions are negated and added to A. Thus, Cl is left only with complements of role assertions. Step 3 iteratively identifies the leaves of subtrees in the graph, groups the assertions concerning the same term into single conjunctive assertion (3a), and further folds them into respective quantification restrictions, reversing the modality obtained in the proof: Skolem terms, introduced by π rule, are turned into universal restrictions (3b), whereas variables, following from ν expansion steps, are replaced with existential restrictions (3c). Notice, that we leave out all the "Skolem" leaves that do not have any qualification, as the only possibility would be to assert $\forall r.\top$ for the predecessors, which is tautologous. In fact these terms belong to branches of the proof that got successfully closed (or resolved). On the other hand we assert $\exists r.\top$ for unqualified variable terms, as in this case we convey a statement of existence of any r-successor. Observe also, that whenever a variable term t_2 is an $r-$successor of another vertex t_1 in the graph, then $\neg r(t_1, t_2)$ is necessarily included in Cl, as it clearly could not have had received a connection in the proof (in such case t_2 would be substituted by an individual name or a Skolem term). Once all the trees are replaced with complex concept assertions about some individuals, the remainder of the solution comprises the set of arbitrary role assertions between known and new individuals, left in Cl.

Like before, consistency checking can be performed via lifted variants of resolution with set-of-support and connection tableaux with restart rule: both, as will be shown in the next section, sound and complete decision procedures for the satisfiability of a τ-transformed DL input with respect to the background DL theory.

Since we have not developed a procedural approach to verifying minimality and relevance of solutions, as well as we have not resolved the problem concerning the role of tautologies in the inference, highlighted in Section 4.3, we are unable to provide a full proof of soundness and completeness of ABox abduction. Instead, we will prove two weaker results, which, as we believe, establish a good starting point for further improvements of the method. For one of them we will require the following lemma.

Lemma 2 ([Loveland, 1978, 2.3.2])**.** *Let S be a minimally unsatisfiable set of clauses, let Cl' be a subset of clause $Cl \in S$, and let $S' = (S \setminus \{Cl\}) \cup \{Cl'\}$;*

i.e. replace Cl in S with Cl'. Then every minimally unsatisfiable subset of S' contains Cl'.

Theorem 7 ($\vdash_{\text{ABox}}$: plain soundness and minimality completeness)**.** *Let $\langle \mathcal{K}, \Phi \rangle$ be an ABox abduction problem in $\mathcal{ALC}$. The following conditions hold:*

1. *If A is a $\vdash_{\text{ABox}}$-plain solution to $\langle \mathcal{K}, \Phi \rangle$ then it is a plain solution to $\langle \mathcal{K}, \Phi \rangle$.*
2. *If A is a minimal, consistent solution to $\langle \mathcal{K}, \Phi \rangle$ then it is a $\vdash_{\text{ABox}}$-plain and consistent solution to $\langle \mathcal{K} \cup Taut, \Phi \rangle$.*

Proof. 1. Consider the method of retrieving a $\vdash_{\text{ABox}}$ solution A from a clause Cl according to its graph $\mathcal{G}$. To demonstrate that A is indeed a plain solution to $\langle \mathcal{K}, \Phi \rangle$ we have to show that there is a refutation proof for $\mathcal{K} \cup A \cup \overline{\Phi}$. For this argument it will be useful to adopt the modal perspective on the proof systems. Observe, that Cl represents a set of consequences entailed by $\mathcal{K} \cup \overline{\Phi}$, at least one of which has to hold for $\mathcal{K} \cup \overline{\Phi}$ to be satisfiable. The literals in Cl apply either to individuals $\overline{a}$ introduced directly in Φ, or to (sets of) individuals accessible from $\overline{a}$ through different paths of roles and modalities — all correctly recorded in $\mathcal{G}$. If we manage to refute all these consequences, then clearly Φ will have to follow. But this is exactly what the solution retrieval procedure does. Observe that all the literals from Cl are initially negated and then restructured to fit into the syntactic form permitted for $\mathcal{ALC}$ ABoxes. The restructuring has to guarantee that original literals are appropriately "allocated" with respect to $\overline{a}$, according to $\mathcal{G}$.

Some of the assertions can be left without a change (assertions about known or some of the abduced individuals). The others, that apply to terms on certain subtrees of $\mathcal{G}$ will be folded into nested formulas. We have to show that folding up succeeds in conveying the right assertions about the right (sets of) individuals. Let $C(t) \in Cl$ be one of the literals, where t is a Skolem term. We need to express that in fact $\overline{C}(t)$, though t is impossible to interiorize in the language. But since Cl and the associated graph are admissible, there had to take place π expansion of $\langle r(t', t) \rangle P_i(\overline{x})$ in the proof, and so there has to be a unique t' and r in the graph such that $r(t', t)$. From the modal viewpoint, this means, that one of the possible consequences of $\mathcal{K} \cup \overline{\Phi}$ is that individual t' has an r-successor, which is C. According to the adopted method we pose $(\forall r.\overline{C})(t')$, which clearly conveys that $\overline{C}(t)$, but can be expressed as an assertion about the predecessor of t, higher on the tree. If, on the contrary, t is a variable on a tree with $r(t', t)$, meaning that every r-successor of t' is C, by the same token we arrive to assertion $(\exists r.\overline{C})(t')$. The procedure is repeated as long as the tree is folded up to the point where the formula can be asserted about a named individual or an individual that is going to be given a new name. By inductive hypothesis, we conclude that nested quantification restrictions formed in this way, successfully refute the consequences that apply to specific (sets of) successors of $\overline{a}$ or other individuals explicitly related to them. Therefore, $\mathcal{K} \cup A$ refutes $\overline{\Phi}$, and so A is a plain solution to $\langle \mathcal{K}, \Phi \rangle$.

2. In this proof we will rest on the FOL perspective on the procedures. In order to do so, we have to acknowledge that both methods can actually

be seen as FOL provers for standard translation of $\mathcal{ALC}$, with some negligible extras. Notice, that presence of modal operators and non-DL predicates does not affect completeness or soundness of the FOL inference for the DL input. Due to character of the transformation τ, and ν and π rules, any clause that would be available in the proof, had we explicitly applied standard translation, Skolemization, and CNF transformation of DL constructs, is still available as a sequence of connections through non-DL predicates and/or ν, π–expansion steps. At the same time it is not possible to close a tableau or derive an empty clause unless it is achievable for the underlying DL theory alone. Mind that negated non-DL predicates occur always in non-unit clauses so using them on a branch or in a resolution step always leaves models that can only be closed with literals from DL's signature.

Now we can roughly follow the proofs of completeness for the propositional case. Let A be a minimal, consistent solution to $\langle \mathcal{K}, \Phi \rangle$. From the minimality assumptions, it follows that no assertion or part of the concept description in an assertion can be removed from A or else it would fail to solve $\langle \mathcal{K}, \Phi \rangle$. Let S_A be a set of clauses obtained from A by the standard translation, Skolemization and CNF reduction. Clearly, $\Theta = \tau(\mathcal{K} \cup \overline{\Phi}) \cup \mathrm{Taut} \cup S_A$ is unsatisfiable and so there exists a mu subset of ground instances of the clauses from Θ. Consider clause $Cl_{\mathrm{init}} = \bigcup \tau(\overline{\Phi})^{\dagger}$. Since A is also a consistent solution, there has to be a ground instance of Cl_{init} that belongs to a mu subset of Θ (by Lemma 1). Therefore, there exists a ground refutation proof (both connection tableau or resolution with set-of-support) for Θ initiated by Cl_{init}. Notice, that unlike before, S_A does not have to come as a set of unit clauses because of universal restrictions, which are turned into disjunctions under standard translation. For that reason we cannot expect anymore that all complements of the literals of the clauses from S_A will appear at some point as leaves of a tableau or as members of a resolvent in a set-of-support. Recall that this has been a key property of the completeness proofs for the propositional case. Still, we can hinge on another observation, which leads to the conclusion that at least certain essential literals will appear on branches.

Notice, that all non-unit clauses in S_A have the form $\{\neg r_1(t_1, t_2), \ldots, \neg r_m(t_{n-1}, t_n), L(\bar{t})\}$, where $L(\bar{t})$ is either a concept literal or a positive role literal. By the minimality assumption there is a mu subset of Θ to which a ground instance of every clause from S_A belongs. Let S_A^G be the set of all those ground instances for some mu subset of Θ, Θ_{Init}. Take a non-unit ground clause $Cl \in S_A^G$ and prune it from all literals except from the last one: the concept or the positive role literal. Replace Cl with the resulting unit clause Cl'. By Lemma 2 we see that Cl' has to belong to every mu subset of Θ_{Init}. Focus on that subset and repeat the operation for every non-unit clause in S_A^G. We now have a set of unit ground clauses $S_A^{G'}$, such that it is contained in some mu subset of Θ_{Init} along with Cl_{init}. Therefore, from here we can continue the proof along the same lines as presented in Sections 3.2 and 3.3 for tableau and resolution procedures respectively. It follows that in any of the two abductive procedures we are able to retrieve clause $Cl_{\overline{A}}$ such that $\sigma(\overline{Cl_{\overline{A}}}) = \bigcup S_A^{G'}$, for

some ground substitution σ. In principle there can be more than one proof resulting in $Cl_{\overline{A}}$. What we have to show, therefore, is that there exists one that allows for a faithful reconstruction of A.

Obviously, the reconstruction is only possible if $Cl_{\overline{A}}$ is a DL-admissible clause, whereas the graph associated with it is $\mathcal{ALC}$-admissible. The existence of such a proof is easy to show. Trivially, in every proof in which unification of $\overline{Cl_{\overline{A}}}$ with $\bigcup S_A^{G'}$ is possible $Cl_{\overline{A}}$ is a DL-admissible clause, or else the unification would not succeed. Also, there has to exist a proof where the associated graph is $\mathcal{ALC}$-admissible as there has to exist a modal (non-connection) proof for the problem.

Focus on a proof where admissibility conditions are satisfied and let us continue the argument. Notice, that all literals in $Cl_{\overline{A}}$ that correspond to the unit clauses in S_A^G can be now clearly given an unambiguous interpretation as they explicitly contain all information sufficient for rendering them back into DL constructs. The only problem concerns the reconstruction of universal restrictions, which are mapped into disjunctions via standard translation. Note that we have confined ourselves to $Cl_{\overline{A}}$ that contains only the last literal from every such disjunction. Recall also that ground instances of these disjunctions belong to Θ_{Init} — the initial mu subset of Θ, for which a connection proof can be constructed. We claim that a proof that properly designates A with respect to all universal restrictions involved in A is one in which all ground instances of non-unit clauses $\{\neg r_1(t_1, t_2), \ldots, \neg r_m(t_{n-1}, t_n), L(\bar{t})\}$ have all inner terms t_2 – t_n in the form of Skolem terms. Notice, that some proofs might have these terms instantiated with individual names instead.[2] Consider for instance $Cl = \{\neg r_1(t_1, t_2), \ldots, r_i(t_k, a), \ldots, \neg r_m(t_{n-1}, t_n), L(\bar{t})\}$, where a is an individual name. Clearly, the retrieval method would collapse on a and fail to interpret the whole nesting of universal restrictions. We will demonstrate, however, that there always exists an alternative mu subset (w.r.t. Θ_{Init}) and so an alternative proof to $Cl_{\overline{A}}$, in which such situation does not occur, everything else remaining the same. Revise Cl in Θ_{Init} to $Cl' = \{\neg r_1(t_1, t_2), \ldots, \neg r_i(t_k, f^\star(t_k)), \ldots, \neg r_m(t_{n-1}, t_n), L(\bar{t})\}$ and add another clause $\{\neg r_i(t_k, a), r_i(t_k, f^\star(t_k))\}$. The clause is derived from a DL tautology $\exists r.\top \sqcap \forall r.C \sqsubseteq \exists r.C$. Clearly, the resulting set Θ'_{Init} is still minimally unsatisfiable as $f^\star$ is a new term in the set. The same treatment should be given to every occurrence of an individual name among the inner terms of non-unit clauses. Eventually, by involving the tautology, we see that there exists a proof in which the universal restrictions will be also reconstructed appropriately.

This completes the proof: for every minimal and consistent solution A to an

[2] In fact, we deal here with a simple case, already signaled in Section 4.3, when $\langle\{r(a, b)\}, \{(\exists r.C)(a)\}\rangle$ is solved by $(\forall r.C)(a)$. Here, the ground instance of the solution that belongs to a mu subset of Θ might be $\{\neg r(a, b), C(b)\}$, which according to the retrieval method cannot be reconstructed into universal restriction, but will lead to assertion $C(b)$ instead. We want to show, therefore, that there is always an alternative mu subset and hence an alternative proof, involving certain tautology, in which the ground instance of solution is $\{\neg r(a, f(a)), C(f(a))\}$. Another approach would be to suitably revise the retrieval method, which in such cases would allow for some indeterminism as for the target form of the reconstruction.

ABox abduction problem there exists a $\vdash_{\text{ABox}}$ proof for the problem, such that it allows for a faithful reconstruction of A. □

Finally, for a small illustration of $\vdash_{\text{ABox}}$ abduction, we will solve the problem of happy John, introduced in Section 4.2. The problem is specified as $\langle \mathcal{K}, \{\textsc{Happy}(\mathsf{John})\}\rangle$, where the knowledge base consists of $\mathcal{T} = \{\textsc{Optimist} \sqcup (\textsc{Nihilist} \sqcap \exists \mathsf{owns}.\textsc{Dog}) \sqsubseteq \textsc{Happy}, \textsc{Ignorant} \sqsubseteq \textsc{Optimist}\}$ and $\mathcal{A} = \{\textsc{Nihilist}(\mathsf{John}), \textsc{Dog}(\mathsf{Snoopy})\}$. We start with the transformation of $\mathcal{K}$ to the computation ready form, as presented in Table 5.7.

$\textsc{Optimist} \sqcup$	$\{\neg\textsc{Optimist}(x_1), \textsc{Happy}(x_1)\}$
$(\textsc{Nihilist} \sqcap \exists \mathsf{owns}.\textsc{Dog})$	$\{\neg\textsc{Nihilist}(x_1), [\mathsf{owns}(x_1, x_2)]P_1(x_1, x_2), \textsc{Happy}(x_1)\}$
$\sqsubseteq \textsc{Happy}$	$\{\neg P_1(x_1, x_2), \neg\textsc{Dog}(x_2)\}$
$\textsc{Ignorant} \sqsubseteq \textsc{Optimist}$	$\{\neg\textsc{Ignorant}(x_1), \textsc{Optimist}(x_1)\}$
$\textsc{Nihilist}(\mathsf{John})$	$\{\textsc{Nihilist}(\mathsf{John})\}$
$\textsc{Dog}(\mathsf{Snoopy})$	$\{\textsc{Dog}(\mathsf{Snoopy})\}$

Table 5.7: Happy John problem: transformation of $\mathcal{K}$.

For parsimony, we will compute one part of solutions using the tableau method and the other part by resolution, though obviously both procedures generate the same answers. The three tableau trees in Figure 5.2 provide three solipsistic solutions, all associated with the same one–vertex graph. As expected, the solutions are: $\textsc{Happy}(\mathsf{John})$ (trivial), $\textsc{Optimist}(\mathsf{John})$ and $\textsc{Ignorant}(\mathsf{John})$.

Figure 5.3 represents a more interesting part of the solution space, explored by the resolution procedure. The search follows a different path than before and encounters a modal operator, which has to be expanded with ν rule. At that stage the generated resolvent has its graph extended, with respect to its parent's graph, with a new vertex x_2 and an edge $\mathsf{owns}(\mathsf{John}, x_2)$. After resolving the non-DL predicate P_1 we obtain two solutions. The solution marked as IV. is retrieved as $(\exists \mathsf{owns}.\textsc{Dog})(\mathsf{John})$, simply by folding the assertions about x_2 into an existential restriction on individual John.[3] The next clause results in role assertion $\mathsf{owns}(\mathsf{John}, \mathsf{Snoopy})$. Notice, that graphs associated with clause V. are different, as in V. vertex x_2 is substituted with individual name Snoopy.

5.3 Consistency checking and blocking

In order to verify consistency of a solution A with respect to the knowledge base $\mathcal{K}$ it is again sufficient to follow a connection-driven search for a refutation

[3]Such a solution, as reported in Section 4.3, is always minimal with respect to one in which we explicitly make assertions about a new individual.

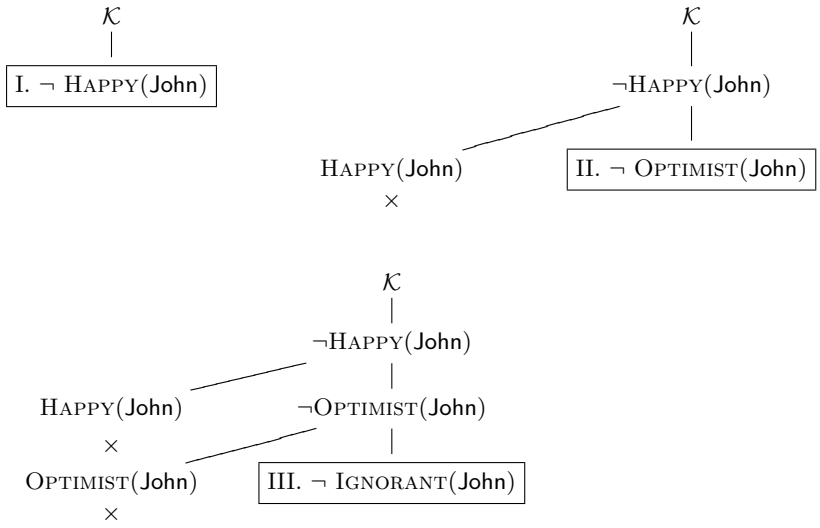


Figure 5.2: Happy John problem: tableaux solutions.

proof of $\mathcal{K} \cup A$. As already argued in the proof of Theorem 7, the two discussed methods can be considered as full FOL procedures for checking satisfiability, and as such guarantee soundness and completeness of the inference. The tableaux procedure should be of course augmented with the restart rule, which restores proof confluency, thus greatly facilitating the process. The knowledge base should be provided in the form $\tau(\mathcal{K}) \cup \tau(A)^{\ddagger}$, whereas the set of initial clauses as $\tau(A)^{\dagger}$.

A significant difference between the use of the procedures for consistency checking and abductive reasoning concerns the possibility of skipping any inference steps, which result in literals that cannot be immediately grounded. In the former case, we are only interested in reasoning about existing or entailed individuals, but not at all on conjectured ones. The refinement can be suitably accounted for as a restriction on the application of ν rule:

Forced role connection Use a ν expansion rule to a literal $[r(t_1, t_2)]P(\overline{x})$ only if $r(t_1, t_2)$ can be immediately connected further.

The refinement corresponds to the typical constraint used in the proof systems for modal logic K, where a $\Box$-formula can be expanded only if there is an immediate successor accessible from the current world. The same condition drives also the inference in all standard DL-reasoners based on the tableau technique, such as FaCT, Pellet and others [Baader and Sattler, 2001]. This suggests a very natural idea of employing the existing reasoners for the deductive tasks that have to be handled in the process of solving an abductive problem. It seems more reasonable to rely on tools that have already been de-

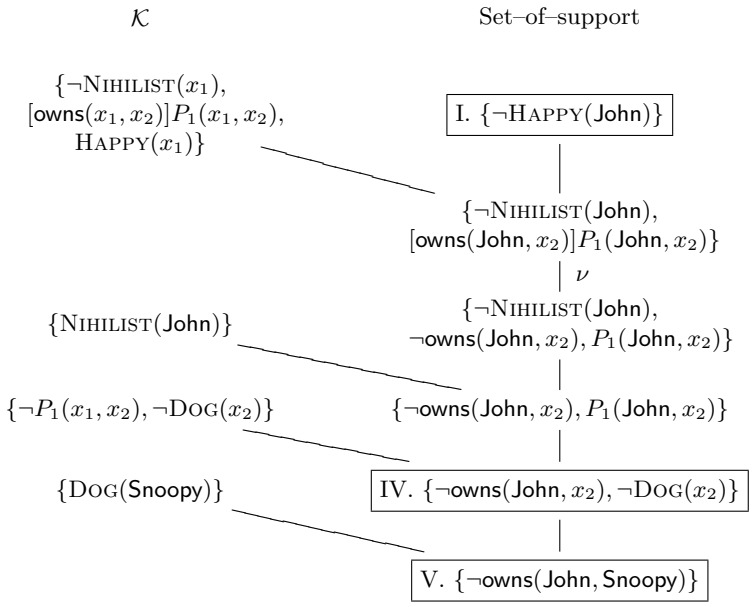


Figure 5.3: Happy John problem: resolution solutions.

veloped through years of intensive research and have a considerably long and successful history of implementations and applications, rather than try reinventing the wheel. In principle there is nothing simpler than that. Given a problem $\langle \mathcal{K}, \Phi \rangle$, where $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, and a plain solution A, we can query for the satisfiability of $\mathcal{A} \cup A$ with respect to the TBox. This form of inference is one of the standard services offered by practically all of the existing reasoners. Alas, we have to sacrifice a crucial benefit offered by our procedures, namely, goal-orientedness of the reasoning. A standard reasoner checks the satisfiability of the whole ABox, also in parts logically not related to the solution A, even though we work under the assumption that the input knowledge base is consistent. Considering that the number of potential solutions to be verified for one problem can be large, the approach is surely inefficient. On the other hand, the reasoning on small problems is likely to be faster in such reasoners, due to lack of unification algorithms and general simplicity of the representation of proofs. It should be expected that the optimal solution can be based on some form of compromise between the two approaches.

Our preliminary proposal is to keep the clause form representation of the knowledge base, while restricting the introduction of terms. Consequently, we save the core idea of connection-driven proof construction, although in a slightly limited scope, while considerably simplifying the selection procedure. The only terms we will be using are individual names for top clauses of ABox axioms and variables for all remaining clauses obtained from the $\pi_\sqcap \circ \pi_\neg$-transformation of

the knowledge base. Hence, the only type of unification that has to be accounted for is substitution of an individual name for a variable.

For any knowledge base $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ and a verified solution A, the following set represents the background knowledge of the inference.

$$\begin{aligned}\Theta := \quad & \{Cl^{x^\star} \,|\, Cl \in \pi_\sqcap \circ \pi_\neg(\mathcal{T})\} \\ \cup \quad & \{Cl^{x} \,|\, Cl \in \pi_\sqcap \circ \pi_\neg(\{C(x) \mid C(x) \in \mathcal{A}\})^\dagger\} \\ \cup \quad & \{Cl^{x^\star} \,|\, Cl \in \pi_\sqcap \circ \pi_\neg(\mathcal{A})^\ddagger\} \\ \cup \quad & \{Cl^{x^\star} \,|\, Cl \in \pi_\sqcap \circ \pi_\neg(A)^\ddagger\}\end{aligned}$$

As the initial input Γ we are going to use the set of top clauses of the solution and all role assertions present in the solution and the ABox.

$$\Gamma := \{Cl^x \mid Cl \in \pi_\sqcap \circ \pi_\neg(\{C(x) \mid C(x) \in A\})^\dagger\} \cup \{r(t_1, t_2) \mid r(t_1, t_2) \in \mathcal{A} \cup A\}$$

The flow of the proof is guided by the set of condition-action rules, which fire freely whenever a specified condition is satisfied. On every application of a rule to a set of literals Γ, the set is replaced with one or several other sets, whose content is determined by the type of the rule that was used. To ease a comparison, the following characterization of the rules employs the typical notation used for presenting tableau-based DL-reasoners (cf. [Baader and Sattler, 2001]).

The $\Rightarrow_\sqcup$-rule
Condition: $C(x) \in \Gamma$ and $\{\overline{C}\} \cup Cl^y \in \Theta$, where x and y are unifiable, but there is no $D_i(x) \in \Gamma$ for any of $D_i \in Cl$.
Action: $\Gamma^i := \Gamma \cup \{D_i(x)\}$ for every $D_i \in Cl$.

The $\Rightarrow_\exists$-rule
Condition: $(\exists r.P)(x) \in \Gamma$ but there is no individual name y such that $r(x, y), P(y) \in \Gamma$.
Action: $\Gamma' := \Gamma \cup \{r(x, y), P(y)\}$ where y is a new individual name in Γ.

The $\Rightarrow_\forall$-rule
Condition: $(\forall r.P)(x), r(x, y) \in \Gamma$ but there is no $P(y) \in \Gamma$.
Action: $\Gamma' := \Gamma \cup \{P(y)\}$.

The $\Rightarrow_{\forall^-}$-rule
Condition: $r(x, y) \in \Gamma$ and $\{\forall r.P\} \cup Cl^z \in \Theta$, where x and z are unifiable, but there is no $(\forall r.P)(x) \in \Gamma$ nor $D_i(x) \in \Gamma$ for any $D_i \in Cl$.
Action: $\Gamma^1 := \Gamma \cup \{(\forall r.P)(x)\}$ and $\Gamma^{i+1} := \Gamma \cup \{D_i(x)\}$ for every $D_i \in Cl$.

The $\Rightarrow_{\forall\circ\exists}$-rule

Condition:	$(\forall r.P_1)(x) \in \Gamma$ and $\{\exists r.P_2\} \cup Cl^y \in \Theta$, where x and y are unifiable, but there is no $(\exists r.P_2)(x) \in \Gamma$
Action:	$\Gamma^1 := \Gamma \cup \{(\exists r.P_2)(x)\}$ and $\Gamma^{i+1} := \Gamma \cup \{D_i(x)\}$ for every $D_i \in Cl$.

The $\Rightarrow_{\bot}$-rule

Condition:	$C(x), \neg C(x) \in \Gamma$
Action:	$\Gamma' := \emptyset$

The $\Rightarrow_{\sqcup}$-rule corresponds to normal branching rule augmented with the connection restriction. Notice, that if Cl is empty, the rule simply deletes the set Γ. The rules $\Rightarrow_{\forall}$, $\Rightarrow_{\exists}$ and $\Rightarrow_{\bot}$ are exactly the same as in DL-reasoners. Finally, $\Rightarrow_{\forall^-}$ and $\Rightarrow_{\forall\circ\exists}$ rules are added to account for other specific connections, in which quantification restrictions can participate. The former allows for connecting a clause containing a universal restriction, provided there is a predecessor of the considered individual and the qualifying concept can be immediately connected. In fact the rule can be derived from the following property of DL with inverse roles: $\top \sqsubseteq \forall r.C \sqcup D$ iff $\top \sqsubseteq C \sqcup \forall r^-.D$. The $\Rightarrow_{\forall\circ\exists}$ rule connects a clause containing existential restriction if its expansion can create a successor to which $\Rightarrow_{\forall}$ rule can be immediately applied. This inference step involves some weakening of the connectedness requirement. Basically, at that stage there is no need to assert $\Gamma^1 := \Gamma \cup \{(\exists r.P_2)(x)\}$ but merely $\Gamma^1 := \Gamma \cup \{r(x, y)\}$ for some new individual name y, as the qualifying concept P_2 might not contribute to the proof at all. However, due to the lack of Skolemization it would not be possible to use $P_2(y)$ later if such a need arose. Apart from this small departure, the procedure is entirely connection-driven and, clearly, tailored much better for possible incorporation into the standard DL-reasoners.

Since the signature of a DL knowledge base is finite it follows that all the discussed procedures are guaranteed to terminate for acyclic general TBoxes. In case of cyclic terminologies, termination can only be ensured by employing so-called blocking rules. To observe how cycles can lead to undecidability consider a simple TBox axiom: $\forall r.A \sqsubseteq A$. Let us transform it into a clause $\{(\exists r.\neg A)(x), A(x)\}$ and try to verify, using any of the methods, whether ABox containing single assertion $\neg A(a)$ is satisfiable with respect to the TBox.

As can be seen in Figure 5.4 the procedure generates an infinite chain of r-successors of a, looping over expansion steps applied to the same clause. Blocking rules are special mechanisms, designed specifically for DLs of different expressiveness, that allow for detecting and handling such anomalies. The standard blocking rule for tableau-based procedures in logic $\mathcal{ALC}$ is defined as follows [Baader and Sattler, 2001]:

Subset blocking: an entailed individual a *blocks* another entailed individual b on branch Γ iff b is a successor of a in Γ (not necessarily immediate) and $\{C \mid C(b) \in \Gamma\} \subseteq \{C \mid C(a) \in \Gamma\}$, i.e. the set of concept assertions about b in Γ is subsumed by the set of concept assertions about a.

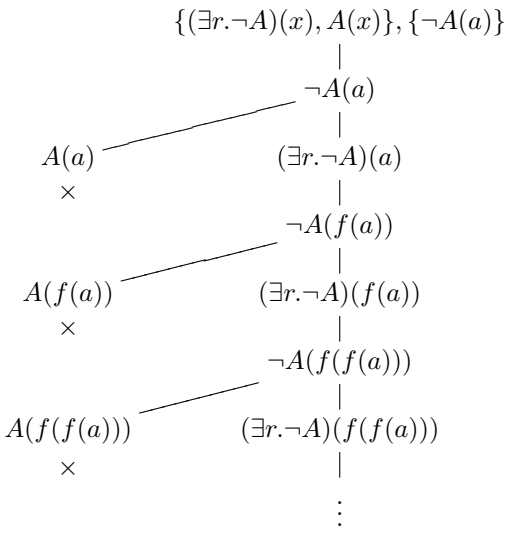


Figure 5.4: Nontermination in cyclic terminologies.

The criterion prevents from expanding existential restrictions on blocked individuals, thus cutting an infinite descent. The intuition behind it should be quite clear: if the set of assertions about b is smaller than those about a then also whatever can be inferred about potential successors of b cannot go beyond the assertions about successors of a, since knowledge in $\mathcal{ALC}$ is propagated only forwards, along property chains. Hence, it is possible to construct a satisfiable model of the branch by looping all relations from b back to corresponding immediate successors of a. According to the rule, individual $f(f(a))$ in Figure 5.4, should be blocked by $f(a)$ (as they both have the same assertions $\{\neg A, \exists r.\neg A\}$) and can be made its own r-successor as a demonstration that the ABox is satisfiable in that setting.

An implicit assumption is that blocking has to be based on the entire knowledge that can be inferred about individuals, or else the resulting model, closed up with a role cycle, might be erroneously believed to be satisfiable. Although connection-based proofs, discussed in this chapter, purposely do not meet this requirement, subset blocking is still a valid method in their context. To see that the condition suffices, notice, that only entailed individuals can block each other. Let a and b be two such individuals, where b is moreover a successor of a. Consider all concept assertions applicable to a and b that can follow from the knowledge base. Clearly, these are only $\neg C \sqcup D$ for every TBox axiom $C \sqsubseteq D$ and C and additionally the assertions propagated from the immediate predecessors of a and b through two types of quantification restrictions:

1. the existential restrictions that entails the individuals on the branch,
2. all universal restrictions that apply to the predecessors of the individuals.

Observe, that TBox axioms equally to both individuals so they cannot in fact bring about a difference in the set of assertions about the individuals. Therefore, blocking can be decided only on the basis of the knowledge propagated through quantification restrictions. But connectedness refinement in no way affects the possibility of making this kind of inferences, as compared to unrestricted tableau methods. Existential restriction is expanded in the same manner ($\Rightarrow_\exists$-rule or π-rule), whereas clauses with applicable universal restrictions in them can be connected to the proof without any obstacles ($\Rightarrow_{\forall^-}$, $\Rightarrow_\forall$-rule or ν-rule). Hence, b is blocked by a in a connection proof if and only if it is blocked by a in a corresponding branch of an unrestricted proof, as only the same assertions can be inferred about the individuals. To ensure that blocking is used only in a sufficiently saturated connection proof, it is necessary to restrict application of the $\Rightarrow_\exists$-rule, so that it could not be triggered as long as other rules apply to the branch. If this condition is satisfied, an existential restriction can be expanded provided it is not blocked. Concluding, the connection-driven tableau-based methods are sound and complete decision procedures for checking consistency of abductive solutions in $\mathcal{ALC}$ under the subset blocking technique.

The problem returns, however, in yet another context. Not surprisingly, cyclic terminologies can also affect termination of the solution finding procedure for abductive problems. For instance, the tableau presented in Figure 5.4 could equally well represent the process of solving problem $\langle\{\forall r.A \sqsubseteq A\}, \{A(a)\}\rangle$, which in this case would return assertions: $(\forall r.A)(a)$, $(\forall r.(\forall r.A))(a)$, $(\forall r.(\forall r.(\forall r.A)))(a)$ and so on. Unlike in case of deduction, abductive reasoning can also involve infinite branches of ν expansion steps, which create infinite chains of abduced individuals. For instance, problem $\langle\{\exists r.A \sqsubseteq A\}, \{A(a)\}\rangle$ can be solved by: $(\exists r.A)(a)$, $(\exists r.(\exists r.A))(a)$, $(\exists r.(\exists r.(\exists r.A)))(a)$.

The issue becomes troublesome once we notice, that all these are in fact minimal and consistent solutions. In practice, termination might not have to be of the first importance for abductive reasoning, nevertheless, solutions containing infinite nestings of the same assertion patterns are also not very interesting, and so it would be desirable to have a reliable technique of excluding them from the solution space, even at the obvious cost of sacrificing completeness to some extent. Although drawing a strict line here is inevitably arbitrary and falls at the scope of pragmatical aspects of abduction rather than logical, it seems quite reasonable to apply the same criterion as before, extended in order to cover blocking of abduced individuals. Hence, we tentatively propose to adopt the following condition:

$\vdash_{\mathbf{ABox}}$ **subset blocking:** a Skolem (variable) term t_1 *blocks* another Skolem (variable) term t_2 on branch Γ in a proof associated with graph $\mathcal{G}$ iff t_2 is a successor of t_1 in $\mathcal{G}$ (not necessarily immediate) and $\{C \mid C(t_2) \in \Gamma\} \subseteq \{C \mid C(t_1) \in \Gamma\}$, i.e. the set of concept assertions about t_2 in Γ is subsumed by the set of concept assertions about t_1.

The criterion guarantees termination of the tableau-based abductive procedure. Given the τ-transformation of the knowledge base there is a finite number

of literals that can be asserted about an individual,[4] hence, there has to be a point where the set of assertions about an entailed or abduced individual has to be subsumed by the set of assertions applying to one of its predecessor. The fact that we cannot impose the prior saturation constraint on application of π and ν rules in this setting, results in the consequence that an individual blocked in one proof might not be so in a different one.

Although we have not studied the behavior of the criterion in the context of abduction in detail, the preliminary observations are quite encouraging. Consider for instance knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, where:

$$\begin{aligned} \mathcal{T} = &\quad \{\forall r.A \sqsubseteq A,\ \exists r.B \sqsubseteq B,\ C \sqsubseteq \forall r.(\forall r.(\forall r.A)),\ A \sqcap B \sqsubseteq D\} \\ \mathcal{A} = &\quad \{(\exists r.(\exists r.(\exists r.B)))(a)\} \end{aligned}$$

and ABox abduction problem $\langle \mathcal{K}, \{D(b)\}\rangle$. The TBox contains two cyclic axioms, which noticeably have to be involved in solving the problem. Some of the consistent solutions that can be found under the blocking technique will be for instance: $\{(\forall r.(\forall r.A) \sqcap \exists r.(\exists r.B))(b)\}$, $\{(C \sqcap B)(b)\}$, $\{A(b), r(b,a)\}$ or $\{C(c), r(c,b), r(b,a)\}$. On the contrary, all those that contain more than two nestings of $\forall r.A$ and $\exists r.B$ will be left out.

Unfortunately, this kind of blocking methods are impossible to mimic in the resolution setting, as there is no direct access to the representation of possible models of the premises during the inference. A naive approach, analyzed in [Hustadt and Schmidt, 1999], is to involve a special form of branching of the set of resolvents, thus obtaining a proof roughly resembling tableau tree. This method, however, makes the very use of resolution totally pointless, and in fact, has been considered only for the purpose of establishing correspondence results between the two reasoning techniques. Inability of maintaining decidability has been one of the reasons why resolution-based procedures for DLs with cyclic TBoxes have not been investigated until just recently [Kazakov and Motik, 2008], and even now require much more sophisticated forms of computation than those discussed in this work.

5.4 Expressive extensions

In the previous sections we have elaborated on the abductive procedures for DL $\mathcal{ALC}$. The results show that both techniques are in principle capable of solving ABox abduction problems in that logic. Though in terms of efficiency the procedures are still far from perfect, we will try, as a final part of our work, to take a glance at some of the expressive extensions of $\mathcal{ALC}$ and make a few remarks on the possibility of covering them in the framework.

Basically, every new feature increasing the expressiveness of $\mathcal{ALC}$ has to be accounted for on three levels: transformation, inference rules and solution retrieval. Also, in case of the tableau procedure, the blocking mechanism has to

[4] We assume every boxed literal $\langle r(t_1, t_2)\rangle P(\overline{x})$ or $[r(t_1, t_2)]P(\overline{x})$ to be a proper assertion about individual t_1, whereas the non-DL predicate $P(\overline{x})$ resulting from its expansion to be an assertion about t_2.

be suitably revised, advisably in the same manner as is commonly accepted in deductive tableaux for the corresponding logics [Baader and Sattler, 2001]. As mentioned in Section 5.2 (see Footnote 2), there is some freedom in the choice of approach to handling the growing expressiveness of the underlying DL in the context of abduction. On the one hand, we might liberate the criteria of admissibility for abductive graphs associated with solutions and focus predominantly on augmenting the solution retrieval method, thus allowing for more indeterminacy in reconstructing ABox assertions from abductive proofs. On the other one, it is possible to keep the retrieval method relatively fixed, while extending the knowledge base with relevant formulas that properly account for additional DL constructs on the level of inference. Whereas the first approach is more parsimonious and should preserve more control over the reasoning, the second one seems to be more uniform, as some additions to the knowledge base are unavoidable also in the former case. Probably a reasonable balance between the two approaches can offer most satisfactory results.

5.4.1 Inverse roles and role hierarchies

Inverse roles, present in DLs marked with symbol $\mathcal{I}$, can be easily handled in abductive proofs by extending a knowledge base with FOL formula $\forall_x \forall_y (r(x,y) \leftrightarrow r^-(y,x))$ (i.e. clauses $\{r(x,y), r^-(y,x)\}, \{\neg r(x,y), \neg r^-(y,x)\}$) for every role name $r \in N_R$. Similarly, role hierarchies expressed via subsumption axioms of the form $r \sqsubseteq s$ in logics $\mathcal{H}$ can be represented by means of formula $\forall_x \forall_y (r(x,y) \rightarrow s(x,y))$ (clause $\{\neg r(x,y), s(x,y)\}$).

For a complete account, it would be essential to devise a more flexible method of connecting both kinds of clauses to proofs. Namely, it has to be possible to reason, in some way, with roles also inside of boxed literals (prior to applying box expansion rules), so that associated graphs can correctly reflect different variants of relational structures.

As a motivating example consider a knowledge base $\mathcal{K}$ in logic $\mathcal{ALCHI}$ with TBox $\mathcal{T} = \{\forall r.C \sqsubseteq A,\ B \sqsubseteq \forall s.C,\ r \sqsubseteq s\}$ and problem $\langle \mathcal{K}, \{A(a)\}\rangle$. Tables 5.8 and 5.9 present two possible proofs, involving reasoning with roles, with corresponding abductive graphs and solutions (for brevity we omit non-DL predicates).

5.4.2 Cardinality restrictions

Cardinality restrictions, introduced in DLs indicated with letter $\mathcal{N}$, should also be relatively straightforward to manage. For an explicit representation it is sufficient to add an axiom $\leq n\ r \sqsubseteq \leq n+i\ r$ for every $i \geq 1$ and any n and r, while suitably normalizing the transformation of all cardinality restrictions occurring in the knowledge base. A more compact approach would require employing additional inference rules that would deal with basic arithmetic reasoning. Also, if we consider a mixture of logic $\mathcal{N}$ with previously discussed $\mathcal{I}$ and $\mathcal{H}$, the framework should account for possibility of reasoning with role axioms inside of cardinality restrictions.

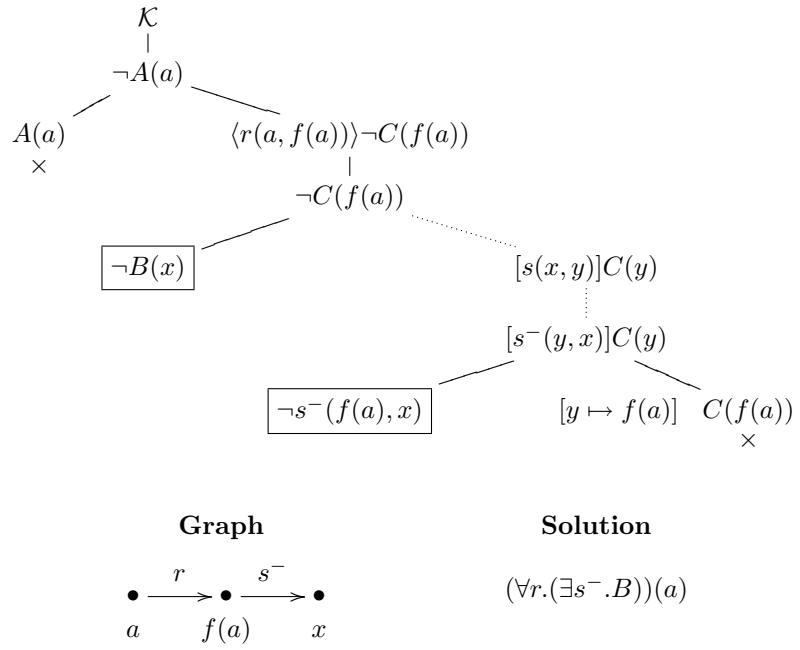


Table 5.8: Abduction with an inverse role.

For instance, problem $\langle\{r \sqsubseteq s\}, \{(\geq 2\ s)(a)\}\rangle$ can be solved by assertions $(\geq 4\ s)(a)$ or $(\geq 2\ r)(a)$. Note, that the first one is non-minimal and generally for any two solutions that differ from each other only by the value of a certain cardinality restriction, one will be non-minimal. Obviously, it is a good news considering there is an infinite set of solutions based on $\geq$-cardinality restrictions.

A bit more complex are logics $\mathcal{Q}$, which allow for using a qualifying concept in a restriction. In this case, the most feasible way of handling the constructs would probably lead through generalization of quantification and cardinality restrictions into a uniform representation and reasoning with them by means of the same set of inference rules. In principle it is possible to interpret the DL quantifiers as a special form of cardinality restrictions. Notice that the following two equivalences have to hold in DL:

- $\exists r.C \equiv \geq 1\ r\ C$
- $\forall r.C \equiv \leq 0\ r\ \neg C$

Consequently, it should be possible to extend and revise ν and π rules in a way that would enable an appropriate treatment of qualified cardinality restrictions in the framework.

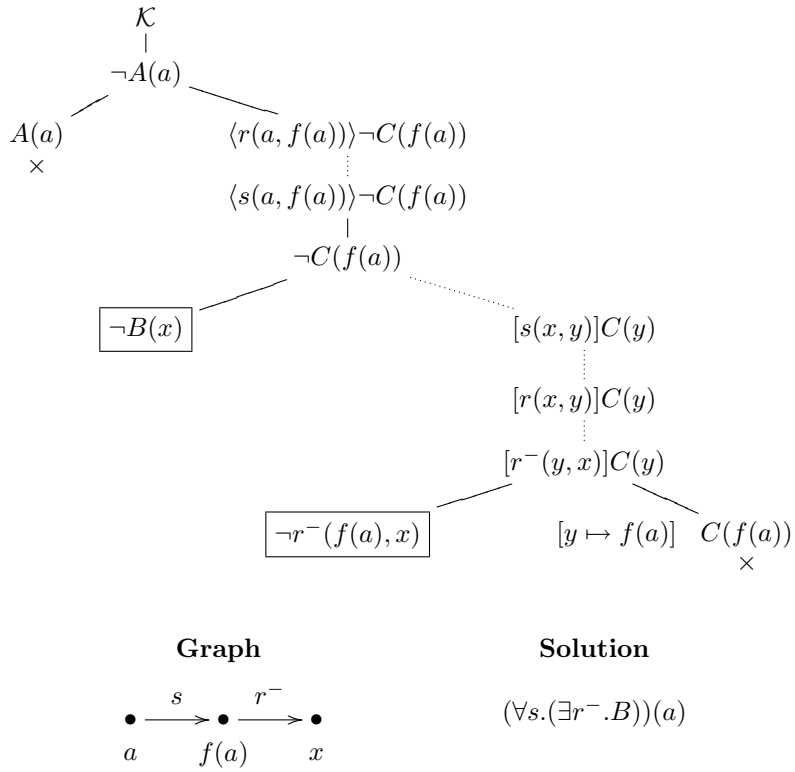


Table 5.9: Abduction with an inverse role and a role subsumption axiom.

5.4.3 Nominals

Nominals are very powerful and, therefore, often desired constructs available in DLs marked with letter $\mathcal{O}$. They allow for a complete internalization of individuals in the language and thus for expressing much more complex relational structures, going beyond tree-like models. Because of that, however, they are also the ones that might be the most difficult to account for. To start with, one should add translations of all assertions in the ABox, obtained according to the following scheme:

- $\{a\} \sqsubseteq C$ for every $C(a)$,
- $\{a\} \sqsubseteq \exists r.\{b\}$ for every $r(a, b)$.

Further, it might be necessary to employ an equality symbol for transforming the new DL axioms into corresponding clauses, which in turn might require more sophisticated reasoning techniques in order to be efficiently handled. Finally,

conditions for admissibility of graphs and the solution retrieval method should be carefully reconsidered as individual names might now occur both as parts of concept descriptions and as terms in the scope of those descriptions. Moreover, there is more syntactic possibilities of expressing certain types of assertions.

For a sample illustration of abductive reasoning with nominals (without equality) consider a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, with $\mathcal{T} = \{\forall r.A \sqcap \exists r.A \sqsubseteq C\}$ and $\mathcal{A} = \{A(b)\}$, and an ABox abduction problem $\langle \mathcal{K}, \{C(a)\}\rangle$. Table 5.10 presents a solution to the problem in which the assertion from the ABox is used twice: once as clause $\{\neg\{b\}(x), A(x)\}$ (the left side of the tableau), the other time as a standard concept assertion $\{A(b)\}$ (right side of the tableau). Notice, that alternatively one could use $\{\neg\{b\}(x), A(x)\}$ in both cases, obtaining solution $(\forall r.\{b\} \sqcap \exists r.\{b\})(a)$ instead.

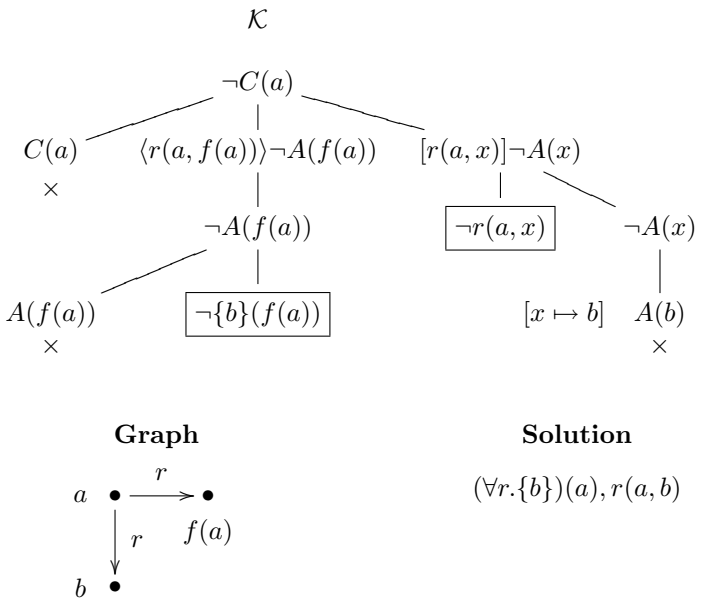


Table 5.10: Abduction with nominals.

Chapter 6

Conclusion

Investigations into abduction are very hard to keep within some strict conceptual and methodological boarders. Logical aspects get easily mixed with computational, philosophical and even pragmatical views on the inference. In many cases it is truly impossible to draw exact lines of separation between the different perspectives.

Our goal in this thesis has been to shed new light on abduction from a *logical standpoint* or, more precisely, from the perspective of *computational logic*, as the research areas embracing certain parts of automated reasoning and Knowledge Representation are sometimes jointly denoted. Regardless of that, however, we have not managed to refrain ourselves completely from letting some epistemological or cognitive considerations interleave with the main logical theme of the work, for instance, when justifying the formal constraints on abduction (Section 2.2), when introducing the principle of goal-oriented reasoning (the beginning of Chapter 3), or when advocating for a complex reasoning architecture for abduction (Section 3.5).

One might claim that the only purely logical sense of abduction is that of *plain abduction*, i.e. an inference to a formula that added to the background knowledge guarantees entailment of the query. However, within such a narrow scope the inference is not especially interesting; to begin with, it is not even nonmonotonic. Nonmonotonicity occurs no sooner than additional constraints on abductive solutions, such as consistency and minimality, are employed. But imposing those requirements immediately calls for extra-logical justifications. Hence, the exciting and absolutely indispensable part of studying abduction, as well as other nonmonotonic and ampliative forms of reasoning, lies exactly in an attempt to bridge the gap between formal tools of expression and informal, but vivid epistemic intuitions, which we, as human reasoners, develop in everyday reasoning tasks. It is impossible to achieve that without any assumptive input, which cannot be justified otherwise but by pointing at its transparency and obviousness.

We regard our work as fitting precisely in such a research scheme. The presented considerations are merely logical consequences of the adopted epis-

temological and computational principles, which we can only hope, should by themselves be universal and uncontroversial enough to be accepted without reservation by the reader.

6.1 Contributions

The basic result of our work, presented in the first part of the thesis, is showing that goal-oriented abductive reasoning can be effectively interpreted as a connection-driven search for refutation proofs of the knowledge base in the union with the negated query, involving enforcement of the proofs' completion. More specifically, we have proved that regular connection tableaux and resolution with set-of-support are sound and complete computation techniques for solving abduction problems in propositional logic, where solutions satisfy the requirements of consistency, minimality and relevance (Theorems 4 and 6). We have argued that these techniques enable much more efficient and better controlled search through the space of possible solutions to an abductive problem. Moreover, we have suggested several ways of further optimizing the procedures. Among others we have proposed a goal-oriented approach to consistency checking, based on the same reasoning techniques, and a goal-oriented procedure for transformation of propositional knowledge bases into the computation ready form. On the practical side of the research, we have implemented and tested a propositional reasoner for solving abductive problems designed along the elaborated principles.

In the second part of the work we have focused on the main objective, which was to adapt and extend the computation framework to the Description Logic $\mathcal{ALC}$. In order to do so, we have defined a new type of transformation of $\mathcal{ALC}$ knowledge bases into a special computation form, which shares some essential properties with Skolemized clauses obtained through standard translation and Conjunctive Normal Form reduction of DL formulas, as well as with modal formulas containing boxed literals. Further, we have introduced a necessary conceptual foundation for ABox abduction in DL (including the notions of a DL-admissible clause and an $\mathcal{ALC}$-admissible abductive graph), and augmented the two reasoning techniques with some additions required in the new context. As the central result, we have proved that the proposed method of finding solutions to ABox abduction problems in $\mathcal{ALC}$ is plain sound and minimality complete (i.e. every selected solution is plain; all consistent and minimal solutions are found, given the knowledge base is extended with the set of DL tautologies) (Theorem 7). The tableau-based procedure, if supplied with blocking rules, is moreover guaranteed to terminate for finite knowledge bases, also with general, cyclic TBoxes. Finally, we have remarked on a possibility of employing standard DL reasoners for an (almost) goal-oriented consistency checking of abductive solutions and indicated potential ways of covering some of the expressive extensions to $\mathcal{ALC}$ in the framework.

Concluding, the goals of the thesis have been to a large extent achieved. We believe that provided results establish a good basis and the first step towards

abductive DL reasoners that could be of practical use in DL-based knowledge systems and Semantic Web applications.

6.2 Discussion and further work

The presented framework for ABox abduction exhibits several shortcomings on the current stage of elaboration. First of all, it is necessary to propose adequate semantic notions of the minimality and relevance constraints for various DLs, along with computation methods that could efficiently apply them. Without them it is impossible to ensure completeness and termination of the procedures. Apart from that, the resolution-based approach requires a suitable blocking mechanism for termination to be guaranteed on cyclic TBoxes.

Second, both methods still demonstrate a significant level of indeterminacy and redundancy in proof construction. We foresee at least two ways of addressing this problem in the future research. One promising strategy involves designing appropriate ordering restrictions for both techniques. Orderings should prevent the same proofs to be obtained through different sequences of inference steps — the problem reported in Section 3.5. Another refinement that should be given attention comes down to eliminating proofs that are expected to reach dead ends, by anticipating upcoming unification steps. Under the proposed transformation it is possible to directly connect to proofs formulas that have been extracted from nested quantification restrictions. However, for a proof like this to succeed it is necessary that the whole superformula that contained the connected clause is unifiable with the proof, resulting in an admissible graph. To support such a verification procedure every TBox and ABox axiom could be associated with the graph representing its tree-shaped model, whereas all the clauses obtained from the transformation of the axiom could be linked to the corresponding vertices of the graph. Inference steps in a proof construction would then be restricted only to such clauses that contain connecting literals and are associated with graphs that can be correctly matched against the abductive graph underlying the entire proof.

The third issue that should necessarily be addressed in the next turn is devising a way of separating to a maximum possible extent reasoning on concepts from reasoning on roles, or at least elaborating a more uniform approach to handle roles. Already now, the inference involving roles lacks some simplicity and becomes really cumbersome when extensions involving more expressive role-related constructs are introduced. Possibly, a bigger share of reasoning on roles can be performed directly on the associated abductive graph rather than in the scope of the proper proof or inside of boxed literals, as provisionally suggested in Section 5.4.1.

Once all the above problems are resolved we should be able to systematically approach other DLs being of a particular application interest, such as $\mathcal{EL}$ (a lightweight DL typically used for large terminologies [Bienvenu, 2008]), $\mathcal{SHOIN}$ (DL underlying OWL 1) or $\mathcal{SROIQ}$ (DL underlying OWL 2, [Horrocks et al., 2006]). The next step towards designing practical DL reason-

ers has to involve tightening the links between the proposed procedures and existing reasoning tools and finally parametrization of the inference in order to support additional, user-specified selection strategies, including preferential criteria, possibility of marking abducibles, or applying specific search heuristics.

6.3 Acknowledgments

I would like to thank Ulle Endriss for providing me with help, academic advice and lots of fantastic teaching all through my studies at the ILLC, as well as for supervising this project. His guidance and support have been invaluable.

I am also very grateful to the people from the Leibniz Center for Law for offering me a remarkable opportunity of getting involved in the work on the Estrella project[1], which has given me a proper start in the field of Knowledge Representation.

[1] `www.estrellaproject.org`

Bibliography

[Aliseda-Llera, 1997] Aliseda-Llera, A. (1997). *Seeking Explanations: Abduction in Logic, Philosophy of Science and Artificial Intelligence*. PhD thesis, ILLC Disertation Series, University of Amsterdam.

[Andrews, 1981] Andrews, P. B. (1981). Theorem proving via general matings. *Journal of the ACM*, 28(2):193–214.

[Areces et al., 2001] Areces, C., de Rijke, M., and de Nivelle, H. (2001). Resolution in modal, description and hybrid logic. *Journal of Logic and Computation*, 11(5):717–736.

[Baader and Nutt, 2003] Baader, F. and Nutt, W. (2003). Basic description logics. In Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., and Patel-Schneider, P. F., editors, *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 47–100. Cambridge University Press.

[Baader and Sattler, 2001] Baader, F. and Sattler, U. (2001). An overview of tableau algorithms for description logics. *Studia Logica*, 69:5–40.

[Bachmair and Ganzinger, 2001] Bachmair, L. and Ganzinger, H. (2001). Resolution theorem proving. In Robinson, A. and Voronkov, A., editors, *Handbook of Automated Reasoning*, pages 1–82. Elsevier Science Publishers B.V.

[Beckert and Goré, 1997] Beckert, B. and Goré, R. (1997). Free variable tableaux for propositional modal logics. In *Proceedings of International Conference on Theorem Proving with Analytic Tableaux and Related Methods, Pont-a-Mousson, France*, volume 1227, pages 91–106. Springer-Verlag.

[Bibel, 1981] Bibel, W. (1981). On matrices with connections. *Journal of the ACM*, 28(4):633–645.

[Bienvenu, 2008] Bienvenu, M. (2008). Complexity of abduction in the EL family of lightweight description logics. In *Proceedings of KR2008, 11th International Conference on Principles of Knowledge Representation and Reasoning*.

[Blackburn and van Benthem, 2006] Blackburn, P. and van Benthem, J. (2006). Modal logic: a semantic perspective. In Blackburn, P., van Benthem, J., and Wolter, F., editors, *Handbook of Modal Logic*, pages 1–82. Elsevier.

[Bonatti et al., 2006] Bonatti, P., Lutz, C., and Wolter, F. (2006). Description logics with circumscription. In *Proceedings of Knowledge Representation 2006 (KR-06)*, pages 400–410.

[Colucci et al., 2004] Colucci, S., Noia, T. D., Sciascio, E. D., Donini, F., and Mongiello, M. (2004). A uniform tableaux-based approach to concept abduction and contraction in ALN. In *Proceedings of the 2004 International Workshop on Description Logics (DL2004)*.

[de Rijke, 1998] de Rijke, M. (1998). Description logics and modal logics. In Franconi, E., Giacomo, G. D., MacGregor, R. M., Nutt, W., and Welty, C. A., editors, *Proceedings of the 1998 International Workshop on Description Logics (DL'98)*.

[Elsenbroich, 2005] Elsenbroich, C. (2005). *Instinct for Detection*. PhD thesis, Department of Computer Science, King's College London.

[Elsenbroich et al., 2006] Elsenbroich, C., Kutz, O., and Sattler, U. (2006). A case for abductive reasoning over ontologies. In Bernardo Cuenca Grau, Pascal Hitzler, C. S. and Wallace, E., editors, *Proceedings of the OWLED'06 workshop on OWL: Experiences and Directions 2006*, volume 216.

[Endriss et al., 2004] Endriss, U., Mancarella, P., Sadri, F., Terreni, G., and Toni, F. (2004). The CIFF proof procedure for abductive logic programming with constraints. In Alferes, J. J. and Leite, J., editors, *Proceedings of the 9th European Conference on Logics in Artificial Intelligence (JELIA-2004)*, volume 3229 of *LNAI*, pages 31–43. Springer-Verlag.

[Flach and Kakas, 2000] Flach, P. and Kakas, A., editors (2000). *Abduction and Induction: Essays on their relation and integration*. Kluwer Academic Publishers.

[Gabbay, 1998] Gabbay, D. (1998). *Elementary Logics: a Procedural Perspective*. Prentice Hall Europe, UK.

[Gabbay and Woods, 2006] Gabbay, D. M. and Woods, J. (2006). Advice on abductive logic. *Logic Journal of the IGPL*, 14(2):182–219.

[Garcia et al., 2007] Garcia, M., Kaplunova, A., and Möller, R. (2007). Model generation in description logics: What can we learn from software engineering? Technical report, Institute for Software, Technology & Systems, Hamburg University of Technology.

[Haarslev and Möller, 1999] Haarslev, V. and Möller, R. (1999). RACE system description. In *Proceedings of DL99, International Workshop on Description Logics, Linköping*, pages 130–132.

[Hähnle, 2001] Hähnle, R. (2001). Tableaux and related methods. In Robinson, A. and Voronkov, A., editors, *Handbook of Automated Reasoning*, pages 101–178. Elsevier Science Publishers B.V.

[Hähnle et al., 2004] Hähnle, R., Murray, N., and Rosenthal, E. (2004). Linearity and regularity with negation normal form. *Theoretical Computer Science*, 328(4):325–354.

[Horrocks, 1998] Horrocks, I. (1998). The FaCT system. In de Swart, H., editor, *Proceedings of the 2nd Int. Conf. on Analytic Tableaux and Related Methods (TABLEAUX'98)*, volume 1397 of *Lecture Notes in Artificial Intelligence*, pages 307–312. Springer.

[Horrocks et al., 2006] Horrocks, I., Kutz, O., and Sattler, U. (2006). The even more irresistible SROIQ. In *Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning*, pages 57–67.

[Horrocks et al., 2003] Horrocks, I., Patel-Schneider, P. F., and van Harmelen, F. (2003). From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26.

[Hustadt and Schmidt, 1999] Hustadt, U. and Schmidt, R. A. (1999). On the relation of resolution and tableaux proof systems for description logics. In Dean, T., editor, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99)*, volume 1, pages 202–207.

[Kakas et al., 1992] Kakas, A. C., Kowalski, R. A., and Toni, F. (1992). Abductive logic programming. *Journal of Logic and Computation*, 2(6):719–770.

[Kazakov and Motik, 2008] Kazakov, Y. and Motik, B. (2008). A resolution-based decision procedure for SHOIQ. *Journal of Automated Reasoning*, 40(2–3):89–116.

[Lorenz, 2004] Lorenz, S. (2004). A tableau prover for domain minimization. *Journal of Automated Reasoning*, 13(3):375–390.

[Loveland, 1978] Loveland, D. W. (1978). *Automated Theorem Proving: A Logical Basis*, volume 6 of *Fundamental Studies in Computer Science*. North-Holland Publishing.

[Mayer and Pirri, 1993] Mayer, M. C. and Pirri, F. (1993). First order abduction via tableau and sequent calculi. *Bulletin of the IGPL*, 1 (1):99–117.

[Mayer and Pirri, 1995] Mayer, M. C. and Pirri, F. (1995). Propositional abduction in modal logic. *Journal of the Interest Group in Pure and Applied Logics*, 3 (6):907–919.

[Möller and Neumann, 2008] Möller, R. and Neumann, B. (2008). Ontology-based reasoning techniques for multimedia interpretation and retrieval. In Kompatsiaris, Y. and Hobson, P., editors, *Semantic Multimedia and Ontologies: Theory and Applications*. Springer.

[Nardi and Brachman, 2003] Nardi, D. and Brachman, R. J. (2003). An introduction to description logics. In Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., and Patel-Schneider, P. F., editors, *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 5–44. Cambridge University Press.

[Paavola, 2004] Paavola, S. (2004). Abduction as logic and methodology of discovery: The importance of strategies. *Foundation of Science*, 9:267–283.

[Paul, 1993] Paul, G. (1993). Approaches to abductive reasoning: an overview. *Artificial Intelligence Review*, (7):109–152.

[Peraldi et al., 2007] Peraldi, S. E., Kaya, A., Melzer, S., Möller, R., and Wessel, M. (2007). Multimedia interpretation as abduction. In *International Workshop on Description Logics (DL-2007)*.

[Popper, 1959] Popper, K. R. (1959). *The Logic of Scientific Discovery*. Hutchinson, London.

[Quine, 1959] Quine, W. V. O. (1959). On cores and prime implicants of truth functions. *The American Mathematical Monthly*, 66(9):755–760.

[Schild, 1991] Schild, K. (1991). A correspondence theory for terminological logics: preliminary report. In *Proceedings of IJCAI-91, 12th International Joint Conference on Artificial Intelligence*, pages 466–471, Sidney, AU.

[Schlobach et al., 2007] Schlobach, S., Huang, Z., Cornet, R., and van Harmelen, F. (2007). Debugging incoherent terminologies. *Journal of Automated Reasoning*, 33(3):317–349.

[Schurz, 2002] Schurz, G. (2002). Models of abductive reasoning. *TPD Preprints*, (1).

[Simon, 1977] Simon, H. (1977). *Models of Discovery*. Boston: D. Reidel.

[Tammet, 1992] Tammet, T. (1992). *Resolution Methods for Decision Problems and Finite-Model Building*. PhD thesis, Göteborg University.

[Thagard, 1988] Thagard, P. (1988). *Computational Philosophy of Science*. Cambridge, MA: MIT Press/Bradford Books.

[Waaler, 2001] Waaler, A. (2001). Connections in nonclassical logics. In Robinson, A. and Voronkov, A., editors, *Handbook of Automated Reasoning*, pages 1487–1578. Elsevier Science Publishers B.V.

Appendix

In the following we present the essential part of the code of the propositional abductive reasoner implemented in JESS. The program is represented as a set of condition–action rules expressed in LISP-like syntax. Basic carriers of data in JESS are *facts*, which can be asserted (by instantiating a fact template), retracted, modified or duplicated. A fact is structured in a similar manner as frames. It has a name and predefined slots containing single or multiple values. The list below presents a sample sequence of JESS facts:

```
<Fact-1> (literal (proposition p)(value TRUE))
<Fact-2> (literal (proposition q)(value FALSE))
<Fact-9> (solution (literals <Fact-1>)(entailment-tree))
<Fact-10> (solution (literals <Fact-1> <Fact-2>)
                    (entailment-tree <Fact-9>))
```

The instructions in the body of a rule are executed whenever the pattern in the head matches particular collection of facts in the current base of asserted facts. The patterns can use single- and multiple-value variables, starting with `?` and `$?` respectively. The variable in front of a fact pattern, e.g. `?sl<-`, binds the unique identifier of the matched fact.

The following two rules implement the binary resolution inference with built-in factoring. The first one addresses the case where one resolvent `(clause (literals ...))` belongs to the knowledge base; the second, where both resolvents belong to the set-of-support.

```
(defrule resolution-against-knowledge-base
    ?sl<-(solution (literals $?e1 ?old-goal $?e2)
                   (entailment-tree $?ent-tree))
    ?cl<- (clause (literals $?ng1 ?old-goal $?ng2))
    =>
    (if (eq (list)(intersection$ (list ?e1 ?e2)(list ?ng1 ?ng2)))
        then (bind ?new-goals (get-complements (list ?ng1 ?ng2)))
             (assert (solution (literals (union$ ?e1 ?new-goals ?e2))
                               (entailment-tree ?sl ?ent-tree)))))

(defrule resolution-against-set-of-support
    ?sl1<- (solution (literals $?h1 ?literal $?t1)
                     (entailment-tree $?et1))
```

```
    (connection ?literal ?neg-literal)
    ?sl2<- (solution (literals $?h2 ?neg-literal $?t2)
                     (entailment-tree $?et2))
    =>
    (if (eq (list)(intersection$
                     (list ?h1 ?t1)(get-complements (list ?h2 ?t2))))
     then (bind ?et3 (intersection$ (list ?sl1 ?et1) (list ?sl2 ?et2)))
          (assert (solution (literals (union$ ?h1 ?t1 ?h2 ?t2))
                            (entailment-tree ?et3)))))
```

Consistency checking is obtained by means of regular connection tableaux computed via the following rule, which addresses every branch of every tableau independently. A fact representing a branch is replaced by one or more facts if expansion is possible and retracted once complementary literals occur in it.

```
(defrule expand-branch-consistently
    ?b<- (branch (tb-added $?ta1 ?add $?ta2)
                 (literals $?lit)
                 (used-clauses $?uc))
         (connection ?add ?neg)
    =>
    (if (not (member$ ?neg ?lit))
     then (modify ?b (tb-added ?ta1 ?ta2))
          (if (not (member$ ?add ?lit))
           then (modify ?b (literals ?add ?lit))
                (bind ?branches (list ?m))
                (bind ?cl-list (get-connected-clauses ?neg))
                (foreach ?clause ?cl-list
                  (if (not (member$ ?clause ?uc))
                   then (bind ?new-branches (list))
                        (bind ?cl-rest
                         (complement$ (list ?neg)
                          (fact-slot-value ?clause literals)))
                        (foreach ?literal ?cl-rest
                          (foreach ?branch ?branches
                            (bind ?d
                              (duplicate ?branch
                                (tb-added (union$
                                  (list ?literal)
                                  (fact-slot-value ?branch tb-added)))
                                (used-clauses ?clause
                                  (fact-slot-value ?branch used-clauses))))
                             (bind ?new-branches (list ?new-branches ?d))))
                         (foreach ?branch ?branches (retract ?branch))
                    (bind ?branches ?new-branches))))
     else (retract ?b)))
```

The next two rules apply the strategy of minimizing the effort of consistency checking by reusing models of already computed solutions and by removing

subsumed branches of a tableau. The reuse of models is performed by reference to the entailment-tree that is generated along the process of solving a problem.

```
(defrule find-partial-models
    ?sl<- (solution (literals $?literals)(entailment-tree ?first $?))
    =>
    (bind ?tableau (get-tableau ?first))
    (foreach ?branch ?tableau
        (duplicate ?branch (solution ?sl)(tb-added ?literals))))

(defrule tableau-pruning
    ?b1<-(branch (solution ?sl)(tb-added $?tb1)(literals $?lit1))
    ?b2<-(branch (solution ?sl)(tb-added $?tb2)(literals $?lit2))
    (test (neq ?b1 ?b2))
    =>
    (if (subsetp (list ?lit1 ?tb1) (list ?lit2 ?tb2)) then (retract ?b2)))
```

Finally, we remove inconsistent solutions, i.e. those that do not have a single tableau branch associated with them, and non-minimal ones.

```
(defrule remove-inconsistent-solutions
    ?sl<-(solution (literals $?))
    (not (branch (solution ?sl)))
    =>
    (retract ?sl))

(defrule remove-non-minimal-solutions
    ?sl1<- (solution (literals $?first))
    ?sl2<- (solution (literals $?second))
    (test (neq ?sl1 ?sl2))
     =>
    (if (subsetp ?first ?second) then (retract ?sl2))
```

The above exposition is only slightly simplified with respect to the original code. One missing element is an ordering of the rule priorities, fixed by salience parameter, required for a coherent functioning of the inference. Also we use additional slots in certain fact templates for conveying extra information regarding, for instance, justification of the inference steps.