Defeasible Inference with Circumscribed OWL Ontologies

Stephan Grimm[1] and Pascal Hitzler[2]

[1] FZI Research Center for Information Technologies at the University of Karlsruhe
stephan.grimm@fzi.de
[2] Institute AIFB, University of Karlsruhe, hitzler@aifb.uni-karlsruhe.de

Abstract. The Web Ontology Language (OWL) adheres to the open-world assumption and can thus not be used for forms of nonmonotonic reasoning or defeasible inference, an acknowledged desirable feature in open Semantic Web environments. We investigate the use of the formalism of circumscriptive description logics (DLs) to realise defeasible inference within the OWL framework. By example, we demonstrate how reasoning with (restricted) circumscribed OWL ontologies facilitates various forms of defeasible inference, also in comparison to alternative approaches. Moreover, we sketch an extension to DL tableaux for handling the circumscriptive case and report on a preliminary implementation.

1 Introduction

Within the Semantic Web community, the Web Ontology Language OWL [17] is well-established as a language for knowledge representation based on description logics (DLs) [1] to semantically annotate web content. However, its strict adherence to the open-world assumption makes it awkward for certain forms of knowledge representation based on nonmonotonic reasoning. In particular, OWL does not allow for any kind of defeasible inference or default reasoning.

Defeasible inferences are drawn based on assumptions about what typically holds. They become invalid whenever evidence for the atypical case is encountered, which makes reasoning nonmonotonic. Defeasible inference is an acknowledged desirable feature in knowledge-based systems with various applications also in the Semantic Web context [10, 8, 11, 18].

Circumscription [13] is a well-known approach to nonmonotonic reasoning that has recently been integrated into the description logic framework in [4]. Circumscriptive description logics allow for various forms of defeasible inference by imposing a local closure on selected concepts through minimisation of their extensions. In contrast to other nonmonotonic extensions to DLs, such as autoepistemic DL [6] or terminological defaults [2], it particularly allows for the handling of unknown individuals introduced through existential quantification, which facilitates the realisation of defeasible inheritance.

In [4], some decidability and complexity results have been achieved for circumscriptive variants of sub-languages of the DL underlying OWL. However, while tableaux algorithms for circumscription in first-order logic have been investigated in the literature (see e.g. [15]), no practical algorithmisation for reasoning with circumscriptive DLs has been proposed so far, and a treatment of such reasoning within the currently successful tableaux calculi used in state-of-the-art DL reasoners is highly desirable. Moreover, there is not much awareness for circumscriptive DLs as a means for nonmonotonic reasoning in the Semantic Web context, and its characteristics for defeasible inferencing with OWL ontologies has not been investigated thoroughly from an ontology engineering perspective.

In this paper, we investigate the use of circumscriptive DLs within the OWL framework for the realisation of various forms of defeasible inference. We motivate the need for defeasible inference by means of an example scenario that is characteristic for an open and uncontrolled Semantic Web environment, picking up the popular domain of pizza delivery used in [19]. We demonstrate how a circumscriptive extension of the OWL language can be used to realise the various forms of defeasible inference with OWL ontologies by means of comprehensive examples. We report on how circumscriptive DLs compare to alternative approaches in terms of the various kinds of defeasible inference and other nonmonotonic features. Moreover, we suggest an extension to the DL tableaux calculus for reasoning with part of OWL in the circumscriptive case (for the details of which we refer to a technical report [7]). In particular, we introduce the notion of a “preference clash” to filter out tableaux branches which do not represent models of a circumscribed knowledge base. Furthermore, we report on a first proof-of-concept implementation, which we have used to verify our examples.

2 OWL and Circumscriptive Description Logic

In this section, we describe the OWL language, giving formal treatment to the fragment covered by our algorithmisation of reasoning with circumscribed ontologies, and we introduce the basic concepts of circumscriptive description logics.

Description Logics and OWL

OWL [17] has been standardised by the W3C consortium as a language for semantic annotation of web content and is widely accepted within the Semantic Web community. Its most recognised variant OWL-DL is based on the description logic (DL [1]) formalism and provides sophisticated knowledge representation and inferencing capabilities. We will use DL syntax throughout the paper as the most compact way of rendering statements in OWL.

The basic elements used to represent knowledge in the description logic formalism are *concepts*, such as *Pizza*, *roles*, such as *topping*, and *individuals*, such as *margarita*. Complex concept expressions can be formed out of these basic elements using *concept constructors* in a nested way. For example, the complex concept $\mathit{Pizza} \sqcap \exists\, \mathit{topping}\,.\mathit{Mozarella} \sqcap \forall\, \mathit{topping}\,.\neg \mathit{Meat}$ denotes the class of all pizzas that have some mozarella topping but no meat toppings.

While the DL underlying OWL-DL is $\mathcal{SHOIN}(\mathbf{D})$, the fragment that we investigate for defeasible reasoning is the logic $\mathcal{ALCO}$, which we introduce formally. $\mathcal{ALCO}$ provides top and bottom concepts, concept conjunction and disjunction, full negation, existential and universal restriction and nominals, where complex concepts C are formed out of atomic concepts A, individuals a_i and roles r according to the following grammar.

$$C \rightarrow A \mid \bot \mid \top \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \exists r.C \mid \forall r.C \mid \{a_1, \ldots, a_n\}$$

An OWL-DL ontology consists of statements that represent the *axioms* of a corresponding DL knowledge base *KB* composed of a *TBox* and an *ABox*. The TBox describes terminological knowledge by inclusion axioms of the form $C_1 \sqsubseteq C_2$ which state subsumption between two concepts C_1 and C_2. For example, the axiom $\mathit{Pizza} \sqcap \exists\, \mathit{topping}.\mathit{Chili} \sqsubseteq \mathit{SpicyDish}$ states that any pizza that has some chili topping is a spicy dish. Another kind of TBox axiom is a concept equivalence of the form $C_1 \equiv C_2$, which is a shortcut for the two inclusions $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_1$. For example, $\mathit{VegetarianPizza} \equiv \mathit{Pizza} \sqcap \forall\, \mathit{topping}.\neg \mathit{Meat}$ states that vegetarian pizzas are exactly those pizzas all of whose toppings are not meat.

The ABox, on the other hand, describes assertional knowledge by assertion axioms of the forms $C(a)$ and $r(a, b)$, which state membership of an individual a to a concept C and association of two individuals a and b via the role r, respectively. For example, the concept assertion axiom $\mathit{Pizza} \sqcap \mathit{SpicyDish}(\mathit{Vesufo})$ states that Vesufo is a spicy pizza, while the role assertion axiom $\mathit{offers}(\mathit{Paolo's}, \mathit{Vesufo})$ says that Paolo's delivery service offers the pizza Vesufo.

The semantics of description logics, and thus that of OWL-DL, is defined in a model-theoretic way by means of *interpretations*. Formally, an interpretation $\mathcal{I}$ consists of a set $\Delta^{\mathcal{I}}$, called the *domain* of the interpretation, and a mapping from the concept, role and individual names, where individuals directly map to elements of $\Delta^{\mathcal{I}}$, concepts map to subsets of $\Delta^{\mathcal{I}}$, and roles map to binary relations over $\Delta^{\mathcal{I}}$, which are called their *extensions*, respectively. Table 1 specifies this semantics technically in form of conditions for the various constructors, which an interpretation $\mathcal{I}$ must satisfy. Intuitively, an interpretation specifies a particular arrangement of objects in the interpretation domain in terms of membership in concept and role extensions. If the arrangement of objects in $\mathcal{I}$ is in accordance with the axioms in a knowledge base *KB* then $\mathcal{I}$ is called a *model* of *KB*. Formally, $\mathcal{I}$ is a model of *KB* if it satisfies the conditions $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, $C^{\mathcal{I}} = D^{\mathcal{I}}$, $r_1^{\mathcal{I}} \subseteq r_2^{\mathcal{I}}$, $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$ for all the various forms of axioms in *KB*, respectively.

$$\begin{array}{c}
\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \quad , \quad \bot^{\mathcal{I}} = \emptyset \quad , \quad A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \quad , \quad r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \quad , \quad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \quad , \quad \{a_1, \ldots, a_n\}^{\mathcal{I}} = \{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\} \\
(\forall\, r.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall b.(a, b) \in r^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\} \\
(\exists\, r.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b.(a, b) \in r^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}
\end{array}$$

Table 1. Model-theoretic semantics for $\mathcal{ALCO}$ constructs.

Even after filtering out those interpretations that do not satisfy all axioms, a knowledge base has in general a multitude of models in which things are interpreted differently. For example, if we did not say in our ontology whether a particular pizza is spicy or not, there are models in which it is and such in which it is not. In such a case, we say that we have incomplete knowledge about the spiciness of pizzas, which is captured by the situation of multiple models. This style of semantics is also referred to as "open-world" semantics, since we assume that we do not have full knowledge about the domain under consideration.

Reasoning with OWL-DL ontologies is based on the standard reasoning tasks defined for a DL knowledge base. Intuitively, *knowledge base satisfiability* checks an ontology for consistency, *concept satisfiability* checks whether a concept can have instances, *instance checking* tests an individual to be an instance of a concept, and *subsumption* tells us whether a concept is in general more specific than another one. In classical DLs all these tasks can be reduced to each other.

Circumscriptive Description Logics

Circumscription [13] is an approach to nonmonotonic reasoning that rests on the principle of minimising the extensions of selected predicates to enforce a local closure of dedicated parts of the domain model. These ideas have recently been incorporated into description logics in [4], yielding a formalism of circumscriptive DLs. For our presentation of defeasible reasoning with OWL ontologies, we present a simplified form of this formalism restricted to parallel concept circumscription in the logic $\mathcal{ALCO}$ (without fixation of concepts or prioritisation among minimised concepts).[3]

Intuitively, minimisation of concepts restricts their extensions to contain only those individuals, for which there is evidence for containment in the extension. For example, if a concept *VegetarianDish* – to represent the class of vegetarian dishes – is minimised, then it is only satisfiable with respect to some knowledge base if this knowledge base contains evidence for the existence of any vegetarian dish. With respect to the empty knowledge base, for example, it is unsatisfiable. Moreover, any individual which cannot be concluded to be a vegetarian dish is automatically derived to be non-vegetarian, i.e. it is an instance of the concept $\neg$ *VegetarianDish*, which is a desirable inferencing behaviour in many situations, e.g. when the ordering of a non-vegetarian dish shall be avoided in a situation with incomplete information about the menu.

The concepts whose extensions are to be minimised are indicated in a *circumscription pattern*, which is a pair $\mathsf{CP} = (M, V)$ where M and V are finite mutually disjoint sets of concept names called the *minimised* and the *varying*

[3] Our intention is to demonstrate that already such a simplified form of circumscription is sufficient to realise some useful features of common-sense reasoning in the Semantic Web context. Hence, we skip more complicated notions such as concept prioritisation and fixation. Moreover, fixed concepts can be simulated by minimising them together with their complements (see [7] for details).

concepts, respectively.[4] Based on a circumscription pattern CP, a preference relation $<_{\mathsf{CP}}$ allows to compare interpretations in terms of their extensions for the minimised concepts indicated in CP. An interpretation $\mathcal{J}$ is *preferred* over an interpretation $\mathcal{I}$, denoted by $\mathcal{J} <_{\mathsf{CP}} \mathcal{I}$, if the extensions of minimised concepts in $\mathcal{J}$ "miss some instances" compared to those in $\mathcal{I}$. Formally, $\mathcal{J} <_{\mathsf{CP}} \mathcal{I}$ holds for two interpretations $\mathcal{J}$ and $\mathcal{I}$ if the following conditions are satisfied:

(i) $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}}$ and $a^{\mathcal{J}} = a^{\mathcal{I}}$ for all individuals a
(ii) there is some $\tilde{A} \in M$ such that $\tilde{A}^{\mathcal{J}} \subset \tilde{A}^{\mathcal{I}}$
(iii) $\tilde{A}^{\mathcal{J}} \subseteq \tilde{A}^{\mathcal{I}}$ for all $\tilde{A} \in M$

Condition (i) says that only interpretations are compared which share the same domain and also the same assignment of individuals. Conditions (ii) and (iii) assure that $\mathcal{J}$ is actually "smaller" than $\mathcal{I}$ in the extension of some minimised concept, while it is not "bigger" in the extension of any other minimised concept.

Given a circumscription pattern CP, the *preferred models* of a knowledge base *KB* are those models of *KB* that are minimal with respect to $<_{\mathsf{CP}}$. Intuitively, those models are preferred that have the least extensions for the minimised concepts. Reasoning in circumscriptive DL is then performed on a *circumscribed knowledge base* $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$, whose models are just the preferred models of *KB* with respect to CP, and reasoning tasks are defined in the standard way.

- *Concept satisfiability*: a concept C is *satisfiable* with respect to $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$ if there exists a model $\mathcal{I}$ of $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$ in which the extension $C^{\mathcal{I}}$ of C is non-empty.
- *Instance checking*: an individual a is an instance of a concept C with respect to $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$, denoted by $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB}) \models C(a)$, if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ holds for all models $\mathcal{I}$ of $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$.
- *Subsumption*: a concept C_1 is subsumed by a concept C_2 with respect to $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$, denoted by $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB}) \models C_1 \sqsubseteq C_2$, if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ holds for all models $\mathcal{I}$ of $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$.

In contrast with other nonmonotonic extensions of DLs, such as autoepistemic DL [6], the reasoning task of knowledge base satisfiability does not play a specific role in the circumscriptive case. The satisfiability of a circumscribed knowledge base is not affected by just modifying its circumscription pattern, which is captured in the following straightforward proposition.

Proposition 1 (knowledge base satisfiability). *Let KB be an $\mathcal{ALCO}$ knowledge base and let CP be a circumscription pattern. Then $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$ is satisfiable if and only if KB is satisfiable.*

[4] In [4], a circumscription pattern is defined as a quadruple $\mathsf{CP} = (\preceq, M, F, V)$ in terms of predicates in general (rather than concepts only) and it has two more components, namely a further set of fixed predicates that are not affected by minimisation and a partial ordering that determines prioritisation among minimised predicates. As we don't use these features, we omit these two elements, and we further restrict the remaining sets to be finite in the context of practical use.

Proof.
$\Rightarrow :$ As $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$ is satisfiable, it has a model, which by definition is a (preferred) model of KB. Hence, KB is also satisfiable.

$\Leftarrow :$ Assume that KB has a model. Due to the finite model property[5], there is a model $\mathcal{I}$ of KB all of whose concept extensions are finite. Thus, the set of models $\mathcal{J}$ of KB with $\mathcal{J} <_{\mathsf{CP}} \mathcal{I}$ is also finite. Hence, there is some model $\mathcal{J}$ in this set that is minimal with respect to $<_{\mathsf{CP}}$, and which is thus a model of $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$. This shows satisfiability of $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$.

3 Defeasible Inferences with Circumscribed Ontologies

A characteristics of much of our knowledge is that it is quite general but not without exception. While in classical (description) logics a statement either holds universally or it does not, we would sometimes rather like to state that something "typically holds". Inferences based on such knowledge are inherently *defeasible*[6], which means that we are willing to retract conclusions drawn on the basis on what typically holds in favour of evidence for the abnormal case.

In this section, we illustrate in which ways the formalism of circumscriptive DLs can be used to state what typically holds and what kinds of defeasible inference can be realised by circumscribing the knowledge in ontologies.

3.1 Running Example

Throughout the section, we use a running example taken from the popular pizza domain introduced in the OWL context in [19]. In our scenario similar to that used in [9], pizza delivery services allow to order pizzas via the web. On their web sites, they describe their offerings in form of semantic annotations formulated in terms of the vocabulary of a general ontology about pizzas. A Semantic Web agent that comes across such annotations might encounter situations in which the properties of a pizza offering are incompletely specified but still needs to reason about them. To better handle such situations of incomplete knowledge, it can be a desirable strategy for an agent to make common-sense conjectures based on assumptions about what typically holds and to perform defeasible inferencing.

Consider the following ontology O_{Pizza} that pizza delivery services are supposed to use as a basis to semantically annotate their offerings.

$$
\begin{array}{rll}
O_{Pizza} = \{ & \mathit{DeliveryService} \sqsubseteq \forall\, \mathit{offers}\,.\mathit{Pizza},\ \mathit{Vegetable} \sqsubseteq \neg \mathit{Meat} & \\
& \mathit{PizzaVerdura} \equiv \mathit{Pizza} \sqcap \forall\, \mathit{topping}\,.\,\mathit{Vegetable} & \\
& \mathit{Pizza} \sqcap \exists\, \mathit{topping}\,.\mathit{Chili} \sqsubseteq \mathit{SpicyDish} & \\
& \mathit{VegetarianDish} \equiv \forall\, \mathit{topping}\,.\neg \mathit{Meat} & \}
\end{array}
$$

It states, for example, that delivery services offer pizzas, that vegetables are not meat, and that pizzas with chili toppings are spicy dishes.

[5] Shown in [12] for $\mathcal{SHOQ}$, which contains $\mathcal{ALCO}$.
[6] http://plato.stanford.edu/entries/reasoning-defeasible/

Using the knowledge in O_{Pizza}, our agent shall reason about annotations under the assumptions that "*pizzas are typically not spicy*" and that "*dishes are only vegetarian if they are said to be*". This is achieved by including an additional axiom yielding the knowledge base

$$\mathit{KB} := O_{Pizza} \cup \{\mathit{Pizza} \sqsubseteq \neg \mathit{SpicyDish} \sqcup \mathit{HotPizza}\}$$

and by using the circumscription pattern

$$\begin{aligned}\mathsf{CP} = (&M = \{\mathit{HotPizza}, \mathit{VegetarianDish}\},\\ &V = \{\mathit{DeliveryService}, \mathit{Vegetable}, \mathit{Meat}, \mathit{PizzaVerdura}, \mathit{SpicyDish}, \mathit{Chili}\})\end{aligned}$$

for reasoning with the circumscribed knowledge base $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$. By minimisation of the concepts for hot pizzas and vegetarian dishes, the agent does not believe that there are such pizzas, unless it encounters direct evidence for it. The additional axiom in KB can then be read as the general rule of pizzas being non-spicy with the exceptional case of them being "hot", which typically does not occur.

In our example, the pizza delivery services of Giovanni, Emilio, Paolo, Alberto and Ernesto provide semantic annotations of their offerings as follows.

$$\begin{aligned}O_{Giovanni} = \{\ & \mathit{PizzaVerdura}(\mathit{Verdura}),\\ & \mathit{DeliveryService}(\mathit{Giovanni's}), \mathit{offers}(\mathit{Giovanni's}, \mathit{Verdura})\ \}\\ O_{Emilio} = \{\ & \mathit{Pizza} \sqcap \mathit{VegetarianDish} \sqcap \mathit{SpicyDish}(\mathit{Vesufo}),\\ & \mathit{DeliveryService}(\mathit{Emilio's}), \mathit{offers}(\mathit{Emilio's}, \mathit{Vesufo})\ \}\\ O_{Paolo} = \{\ & \mathit{PizzaVerdura} \sqcap \exists\, \mathit{topping} . \mathit{Chili}(\mathit{Diabolo}),\\ & \mathit{DeliveryService}(\mathit{Paolo's}), \mathit{offers}(\mathit{Paolo's}, \mathit{Diabolo})\ \}\\ O_{Alberto} = \{\ & \mathit{DeliveryService} \sqcap \forall\, \mathit{offers} . \mathit{VegetarianDish}(\mathit{Alberto's}),\\ & \exists\, \mathit{offers} . (\mathit{Pizza} \sqcap \mathit{SpicyDish})(\mathit{Alberto's})\ \}\\ O_{Ernesto} = \{\ & \mathit{Pizza} \sqsubseteq \mathit{SpicyDish}, \mathit{DeliveryService}(\mathit{Ernesto's})\ \}\end{aligned}$$

We will look at how the agent reasons about these annotations in both the classical case using KB and in the circumscriptive case using $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$, considering the various standard reasoning tasks introduced in Section 2.

3.2 Defeasible Retrieval

Retrieval asks for the known instances of a given concept and can be reduced to the reasoning task of instance checking. Our agent could want to query annotations for offerings of dishes that are not spicy by retrieving the instances of the concept $\neg \mathit{SpicyDish}$. However, not all delivery services have indicated whether the pizzas they offer are spicy or not. From Giovanni's annotation, for example, it does not classically follow that pizza Verdura is non-spicy.

$$\mathit{KB} \cup O_{Giovanni} \not\models \neg \mathit{SpicyDish}(\mathit{Verdura})$$

"Switching on" the agent's common-sense assumption that "*pizzas are typically not spicy if not said otherwise*", however, we get a different result, and reasoning with the circumscribed knowledge yields that pizza Verdura is non-spicy.

$$\mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Giovanni}) \models \neg \mathit{SpicyDish}(\mathit{Verdura})$$

As there is no evidence in $\mathit{KB} \cup O_{Giovanni}$ for pizza Verdura being spicy, it is not in the extension of the minimised concept *HotPizza* and is thus concluded to be an instance of $\neg$*SpicyDish* due to the additional inclusion axiom in KB.

Conclusions derived on the basis of such assumptions can be *defeated* by adding assertions without causing a contradiction, whereas in a purely classical setting adding counter-evidence for a conclusion results in inconsistency.

One kind of defeat is to directly contradict the defeasible conclusion by asserting its contrary. From Emilio's annotation, for example, it cannot be derived that Vesufo is non-spicy although it is a pizza, too.

$$\mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Emilio}) \not\models \neg \mathit{SpicyDish}(\mathit{Vesufo})$$

As in O_{Emilio} Vesufo is directly asserted to be a spicy pizza, it must be in the extension of the concept *HotPizza* despite this concept being minimised.

Another kind of defeat is to indirectly contradict the defeasible conclusion by deriving counter-evidence from what has been stated. From Paolo's annotation, for example, pizza Diabolo cannot be derived to be non-spicy.

$$\mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Paolo}) \not\models \neg \mathit{SpicyDish}(\mathit{Diabolo})$$

Since in O_{Paolo} pizza Diabolo is stated to have a chili topping, it is inferred to be a spicy dish from the knowledge in O_{Pizza}.

3.3 Defeasible Incoherency

An ontology is incoherent if it contains concepts that cannot have any instances, which is verified by the reasoning task of concept satisfiability. Our agent could want to check whether a dish can be spicy and vegetarian at the same time according to a particular annotation. With respect to Giovanni's annotation, the concept *SpicyDish* $\sqcap$ *VegetarianDish* is classically satisfiable, however, it becomes unsatisfiable once we take the agent's assumptions about spicy pizzas and vegetarian dishes into account by reasoning with the circumscribed knowledge.

$$\mathit{SpicyDish} \sqcap \mathit{VegetarianDish} \ \text{ is unsatisfiable w.r.t } \ \mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Giovanni})$$

The only individual for which there is evidence to be a vegetarian dish is pizza Verdura, which can be concluded to be non-spicy, as we have seen before. Hence, within Giovanni's annotation no dish can be both spicy and vegetarian, although the two concepts *SpicyDish* and *VegetarianDish* remain satisfiable for themselves.

Again, such inference can be defeated in various ways. One way is to directly defeat the unsatisfiability of a concept by the explicit assertion of an individual. In Emilio's offers, pizza Vesufo is said to be both spicy and vegetarian, which makes the concept satisfiable with respect to Emilio's annotation.

$$\mathit{SpicyDish} \sqcap \mathit{VegetarianDish} \ \text{ is satisfiable w.r.t } \ \mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Emilio})$$

The explicit assertion of the individual *Vesufo* to both the concepts *SpicyDish* and *VegetarianDish* clearly gives evidence for an overlap of the concepts' extensions.

A more indirect evidence for a dish that is spicy and also vegetarian is given by pizza Diabolo in Paolo's offering, making the concept satisfiable, too.

$$\mathit{SpicyDish} \sqcap \mathit{VegetarianDish} \ \text{ is satisfiable w.r.t } \ \mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Paolo})$$

Here, pizza Diabolo is concluded to be both vegetarian, as it is of type *PizzaVerdura* having only vegetable toppings, and spicy, as it has a chili topping.

Moreover, unsatisfiability of a concept can even be defeated by evidence for the existence of unknown individuals being instances of the concept. Alberto, for example, used the ontology O_{Pizza} in a different way by not modelling pizza offerings in form of explicit individuals. Instead, the knowledge in Alberto's annotation imposes the existence of some spicy vegetarian pizza offered by Alberto, which also makes the concept satisfiable.

$$\mathit{SpicyDish} \sqcap \mathit{VegetarianDish} \ \ \textsf{is satisfiable w.r.t} \ \ \mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Alberto})$$

As Alberto states to offer some spicy pizza, there is some individual in the extension of the concept *SpicyDish* in all models of $\mathit{KB} \cup O_{Alberto}$, although it is not named. Since all offerings of Alberto are also said to be vegetarian dishes, this unknown individual is always in the extension of the concept *VegetarianDish*, and hence the conjunction of the two concepts is satisfiable despite minimisation.

3.4 Defeasible Inheritance

Automated classification of concepts in inheritance hierarchies is the most ubiquitous application of description logics and is realised by the reasoning task of concept subsumption. Our agent could want to classify concepts for dishes offered according to some particular annotation, to also report on inferred subsumptions at the level of intensional knowledge. For example, it could want to ask whether pizzas are classified under non-spicy dishes when a certain situation of offerings is considered. However, the derivation of inclusion axioms in a classical setting is not dependent on a particular situation of individuals and their concept membership, and from Giovanni's annotation it does not follow that pizzas are non-spicy in general by using conventional reasoning methods.

$$\mathit{KB} \cup O_{Giovanni} \not\models \mathit{Pizza} \sqsubseteq \neg \mathit{SpicyDish}$$

Nevertheless, in the particular situation of Giovanni's offers the exceptional hot pizza case does not occur, so that pizzas do become a subclass of non-spicy dishes when activating the agent's assumptions by reasoning with circumscription.

$$\mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Giovanni}) \models \mathit{Pizza} \sqsubseteq \neg \mathit{SpicyDish}$$

The only pizza offered by Giovanni is Verdura, which cannot be concluded to be spicy. Hence, there is no counter-evidence to the assumption that pizzas are non-spicy and the general subsumption statement can thus be derived.

Also inheritance derived based on assumptions can be defeated in various ways, similar to incoherence. As subsumption states full containment of one concept's extension in another one's on a general level, it can be defeated by giving evidence for some individual being in the extension of the subconcept but not in that of the superconcept. One way to achieve this is, again, by explicit assertion of an individual to the subconcept and to the complement of the superconcept, which is the case in Emilio's annotation for pizza Vesufo.

$$\mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Emilio}) \not\models \mathit{Pizza} \sqsubseteq \neg \mathit{SpicyDish}$$

Here, the presence of the spicy pizza Vesufo gives direct counter-evidence for the agent's assumption that pizzas are typically non-spicy in general.

As before, an indirect way of defeat is by deriving counter-evidence. From Paolo's annotation it can be concluded that pizza Diabolo is spicy as it has Chili on it, which also rules out the subsumption of pizzas being non-spicy in general.

$$\mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Paolo}) \not\models \mathit{Pizza} \sqsubseteq \neg \mathit{SpicyDish}$$

Furthermore, also inheritance can be defeated by anonymous individuals. As before, from Alberto's annotation the existence of some unknown individual can be derived that is a pizza and spicy at the same time, which gives sufficient counter-evidence to the general subsumption.

$$\mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Alberto}) \not\models \mathit{Pizza} \sqsubseteq \neg \mathit{SpicyDish}$$

Observe that in different models of $\mathit{KB} \cup O_{Alberto}$ the spicy pizza offered by Alberto does not necessarily need to be the same individual, but it still needs to exist such that there is no preferred model without any spicy pizza.

Finally, inheritance can also be defeated on the intensional level by the mere addition of an inclusion axiom, without introducing any individuals. Ernesto, for example, states, with respect to his offerings, that all pizzas are spicy dishes, which also defeats the general subsumption between pizzas and non-spicy dishes.

$$\mathrm{circ}_{\mathsf{CP}}(\mathit{KB} \cup O_{Ernesto}) \not\models \mathit{Pizza} \sqsubseteq \neg \mathit{SpicyDish}$$

In case of Ernesto's annotation, there are even no individuals involved at all in producing counter-evidence. Observe that if there were individuals involved then this kind of defeat would coincide with one of those mentioned earlier, using the inclusion axiom to just derive another concept membership.

3.5 Relation to Other Nonmonotonic Extensions of DLs

In the literature on nonmonotonic extensions to DLs, two other approaches have been investigated. The formalism of autoepistemic DL [6, 5, 14] incorporates epistemic operators as an additional language construct to express what a knowledge base "knows" in an introspective way. An epistemic operator **K** refers to classes of things that are "known" to have certain properties, while another operator is related to negation-as-failure. As an alternative approach, terminological defaults [2] result from incorporating default rules, known from default logic, into the DL framework. Such defaults provide a form of concept inclusions that allow for exceptions, and research on them has recently been picked up again for implementation in the state-of-the-art DL reasoner Pellet in [20].

Both the autoepistemic and default extensions to DLs are restricted to reasoning with explicitly named individuals only, which makes them awkward for certain forms of defeasible inferencing presented here. Defeasible retrieval can in principle be realised with those approaches, as known individuals are particularly asked for. In a similar scenario defeasible retrieval based on epistemic operators has been demonstrated in [9]. Also defeasible incoherence can be realised at least with autoepistemic DLs, as demonstrated e.g. in [10] and [8]. However, this approach does not allow for any defeat that is based on the mere existence of unknown individuals. Analogously, none of the approaches restricted to reasoning with known individuals can be used to realise defeasible inheritance. The reason is that inclusion axioms cannot be derived on the basis of conclusions for known individuals only, simply because subsumption affects all individuals, also the unknown ones.

On the other hand, circumscriptive DLs can also principally not be used for some other forms of nonmonotonic reasoning that have been investigated in the context of alternative approaches. One feature that has been realised with epistemic operators are *integrity constraints* [6], which allow for the formulation of axioms as restrictions on known individuals, and whose violation leads to an unsatisfiable knowledge base. As a direct consequence of Proposition 1 from Section 2, constraint violation cannot be indicated by knowledge base unsatisfiability in circumscriptive DLs, since circumscription alone does not affect the satisfiability of a knowledge base.

Another feature that has been realised with epistemic operators are *epistemic queries* [5, 14], which are queries that contain epistemic operators posed to a conventional, non-epistemic knowledge base. While restricting expressiveness, this separation of queries from the knowledge base yields a gain in retrieval performance. In circumscriptive DLs, however, such a separation cannot be established, since the indication for minimisation of concepts in a circumscription pattern applies universally to any occurrence of the concept in both the query and the knowledge base.

4 DL Tableaux Extension for Circumscription

In this section, we sketch an extension of the DL tableaux calculus to handle the circumscriptive case. We sketch an algorithm for reasoning with circumscribed knowledge bases and describe a prototypical implementation, while details are given in the technical report [7].

Algorithmisation

Most state-of-the-art DL reasoners, such as Pellet[7], Racer[8] or FaCT[9], are based on tableaux procedures that try to build a model for a concept with respect to a knowledge base. Such algorithms decompose the complex constructs in the axioms into simpler ones to finally yield atomic assertions. For certain constructs, such as disjunctions, there are several ways of decomposition, and the algorithm branches into non-deterministic choices, constructing a tableaux tree. A fully decomposed branch in this tree represents a class of models that can be constructed from the atomic assertions in the branch, unless it contains a *clash*, which represents a direct contradiction. (For details on DL tableaux procedures see for example [3].)

[7] http://www.mindswap.org/2003/pellet/
[8] Meanwhile RacerPro – http://www.racer-systems.com/
[9] Meanwhile FaCT++ – http://owl.man.ac.uk/factplusplus/

Algorithm 1 Construct a knowledge base KB'.

```
Require: a tableaux branch $\mathcal{B}$ produced for an initial $\mathcal{ALCO}$ knowledge base $\mathit{KB}$
    circumscribed with a circumscription pattern $\mathsf{CP} = (M, V)$
 1: $D := \{\bot\}$, $\mathit{KB}' := \mathit{KB}$
 2: for all $\tilde{A} \in M$ do
 3:   if there are assertions $\tilde{A}(a_1), \ldots, \tilde{A}(a_n) \in \mathcal{B}$ then
 4:     $\mathit{KB}' := \mathit{KB}' \cup \{\tilde{A} \sqsubseteq \{a_1, \ldots, a_n\}\}$
 5:     $D := D \cup \{\{a_1, \ldots, a_n\} \sqcap \neg\tilde{A}\}$
 6:   else
 7:     $\mathit{KB}' := \mathit{KB}' \cup \{\tilde{A} \sqsubseteq \bot\}$
 8:   end if
 9: end for
10: $\mathit{KB}' := \mathit{KB}' \cup \{(\bigsqcup_{D_{\tilde{A}} \in D} D_{\tilde{A}})(\iota)\}$, with $\iota$ a new individual
```

In the circumscriptive case, detection of "normal" clashes is not sufficient to filter out invalid branches, since it might be the case that none of the models represented by a clash-free branch is actually a preferred one. To this end, we introduce the notion of a *preference clash* to additionally test whether a branch represents non-preferred models only.

Definition 1 (preference clash). *Let $\mathcal{B}$ be a tableaux branch constructed with respect to a circumscribed $\mathcal{ALCO}$ knowledge base $circ_{\mathsf{CP}}(\mathit{KB})$. $\mathcal{B}$ contains a* preference clash *if the $\mathcal{ALCO}$ knowledge base KB', constructed according to Algorithm 1, is satisfiable.*

A preference clash in a tableaux branch indicates that the construction of this branch has not led to finding a model for the case of a circumscribed knowledge base, as none of the models represented by the branch is preferred – the respective tableaux procedure needs to further expand the tableaux tree at other places, if there are any left. Detection of a preference clash is reduced to a classical satisfiability test for a knowledge base KB', according to Definition 1.

The knowledge base KB', constructed from the original circumscribed knowledge base $\mathrm{circ}_{\mathsf{CP}}(\mathit{KB})$ and the tableaux branch $\mathcal{B}$ according to Algorithm 1, is initialised with the axioms in KB in line 2, to ensure that models of KB' are also models of KB. Then, the extensions of all minimised concepts $\tilde{A}$, determined by positive assertions $\tilde{A}(a_i)$ of individuals a_i in $\mathcal{B}$, are used as an upper bound for the models of KB', by adding appropriate inclusion axioms in line 5. Any such inclusion axiom ensures the extension of a minimised concept to contain at most the individuals asserted in $\mathcal{B}$, reflecting condition (iii) defined for the relation $<_{\mathsf{CP}}$. If there are no positive assertions then the respective minimised concept is restricted to be empty in line 8. Finally, also condition (ii) is encoded into the axioms of KB' by including the disjunctive concept assertion in line 11, which introduces a new individual ι that does not occur in KB. For at least one of the disjuncts, ι is unified with some individual a_i in the nominal constructs collected in line 6, ensuring that for some minimised concept at least one individual of those asserted in $\mathcal{B}$ is actually not in the extension.

By this, the models of KB' are preferred over all models represented by $\mathcal{B}$ with respect to $<_{\mathsf{CP}}$. Hence, the existence of a model for KB' falsifies any model represented by $\mathcal{B}$ to be preferred, and thus indicates a preference clash in $\mathcal{B}$.

However, to ensure completeness of the extended tableaux procedure, possible equality of newly introduced individuals (e.g. by existential quantifiers) with individuals already known needs to be taken into account. For example, if a new individual x is introduced in a branch due to existential quantification, the least model with respect to $<_{\mathsf{CP}}$ might be one in which x is unified with some individual a_i that occurs in the nominal constructs used in the lines 5 and 6. To handle this case, we introduce an additional expansion rule to the underlying standard tableaux procedure, which creates a new branch for any unification of a new individual with an already existing one. The details of the full tableaux procedure can be found in the technical report [7].

A similar idea has been applied in [15] for circumscriptive reasoning in first-order logic, where a tableaux for first-order formulas in clausal form was presented. However, this calculus does not directly yield a decision procedure for reasoning with DLs as it is only decidable if function symbols are disallowed, which correspond to existential restrictions in DLs. Unlike earlier approaches such as [16], the work in [15] and ours have in common that only a single tableaux branch needs to be kept in memory at a time during computation.

Implementation and Ideas for Optimisation

We have prototypically implemented the aforementioned tableaux procedure for reasoning in circumscriptive $\mathcal{ALCO}$, to test our examples from Section 3. The prototype is implemented in Java and runs as a server application supporting the DIG[10] interface for DL reasoning tasks. In this way, it can e.g. communicate with the Protégé[11] ontology editor and provides reasoning services with (restricted) circumscribed OWL-DL ontologies. As OWL has no means to express circumscription patterns, however, minimised concepts are determined by their name: any concept whose name starts with the string "`Ab_`" is being minimised.[12]

The prototype, available for download[13], is a first direct implementation of the preferential tableaux procedure without any optimisations and can be used for testing small examples only. The worst-case complexity of reasoning with circumscriptive DLs has been shown to be NExp^{NP} in [4], and experiences with our prototype confirm that a straightforward implementation does not yield a practically feasible performance. Instead, optimisation techniques are required, and it remains to be investigated whether known DL tableaux optimisations (see e.g. [1, Chapter 9]) apply in the same way in the circumscriptive case to cope with the high complexity of reasoning with this formalism in practice. Moreover, there

[10] http://dl.kr.org/dig/interface.html
[11] http://protege.stanford.edu/
[12] In the circumscription literature, minimised predicates are often called "abnormal" as part of circumscriptive abnormality theories – hence the abbreviation "`Ab_`".
[13] http://www.fzi.de/downloads/wim/sgr/CircDL.zip

can be optimisations that are specific to preferential tableaux and minimisation. First ideas of such specific optimisations comprise the following:

- *model caching for KB'* – An observation from our experiments is that the content of KB' is identical over many cases of preference clash checks. Hence, techniques of model caching can be employed to avoid repeated model construction for identical versions of KB'.
- *postponing non-minimal assertions* – Another idea is to avoid the construction of non-minimal models affecting the order in which tableaux expansion rules are applied. The assertion of individuals to minimised concepts could be postponed, such that branches with the least extensions of minimised predicates are completed first.
- *early closing of non-preferred branches* – Preference clash checks can either be performed after completion of a branch or any time a branch is modified. The latter case allows for additional optimisations by closing a branch early due to non-minimality, even if in classical tableaux it would be further expanded, which was already pointed out in [15]. In our first experiments, for either strategy of detecting preference clashes examples could be found in which the alternative strategy was outperformed. Hence, also the scheduling of preference clash checks provides room for optimisation.

5 Summary and Outlook

In this paper, we have investigated the use of circumscriptive DLs for the realisation of defeasible inference with OWL ontologies, restricted to the logic $\mathcal{ALCO}$. By means of examples, we have demonstrated how reasoning with circumscribed knowledge bases results in various forms of defeasible inference that are useful in a Semantic Web context. We have also pointed out the primary difference of such inferencing to alternative approaches as their restriction to reasoning with known individuals only, which disallows certain kinds of defeat as well as defeasible inheritance. Furthermore, we have sketched an extension to DL tableaux algorithms that includes detection of preference clashes to find preferred models of a circumscribed knowledge base, and we have reported on a preliminary implementation and first ideas for performance optimisation.

As future work, it remains to extend our algorithm to also cover other language constructs of OWL, such as number restrictions, and also more complex features of circumscription, such as prioritisation among minimised concepts. As detection of preference clashes constitutes only a slight modification of conventional tableaux algorithms, it should easily integrate with current optimised state-of-the-art DL reasoners, and it would be interesting to see how well their built-in optimisations apply to the circumscriptive case in terms of performance. Moreover, optimisations specific to minimisation need to be devised on top of these. Furthermore, a methodology remains to be developed for the formulation of appropriate circumscription patterns in various cases of defeasible inferencing, in particular for the more complicated notions of prioritisation and fixation not addressed in this paper.

References

1. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.
2. F. Baader and B. Hollunder. Embedding Defaults into Terminological Representation Systems. *J. Automated Reasoning*, 14:149–180, 1995.
3. F. Baader and U. Sattler. Tableau Algorithms for Description Logics. In *Proc. of Intern. Conf. on Aut. Reas. with Tableaux and Related Methods*, pages 1–18, 2000.
4. P. Bonatti, C. Lutz, and F. Wolter. Expressive Non-Monotonic Description Logics Based on Circumscription. In *Proc. of 10th Intern. Conf. on Principles of Knowledge Representation and Reasoning (KR'06)*, pages 400–410, 2006.
5. F. M. Donini, M. Lenzerini, D. Nardi, W. Nutt, and A. Schaerf. An Epistemic Operator for Description Logics. *Artif. Intell.*, 100(1-2):225–274, 1998.
6. F. M. Donini, D. Nardi, and R. Rosati. Description Logics of Minimal Knowledge and Negation as Failure. *ACM Transactions on Comput. Logic*, 3(2):177–225, 2002.
7. S. Grimm and P. Hitzler. Reasoning in Circumscriptive $\mathcal{ALCO}$. Technical report, FZI at University of Karlsruhe, Germany, September 2007. Available at http://www.fzi.de/downloads/wim/sgr/circDL.pdf.
8. S. Grimm and P. Hitzler. Semantic Matchmaking of Web Resources with Local Closed-World Reasoning. *Int. Journal of eCommerce (IJEC)*, 12(2):89–126, 2008.
9. S. Grimm and B. Motik. Closed-World Reasoning in the Semantic Web through Epistemic Operators. In *CEUR Proc. of the OWL ED Workshop*, 2005.
10. S. Grimm, B. Motik, and C. Preist. Matching Semantic Service Descriptions with Local Closed-World Reasoning. In *Proceedings of the 3rd European Semantic Web Conference (ESWC'06)*, pages 575–589, 2006.
11. V. Kolovski, B. Parsia, Y. Katz, and J. Hendler. Representing Web Service Policies in OWL-DL. In *Proc. of the 4th Int. Sem. Web Conf. (ISWC)*, pages 461–475, 2005.
12. C. Lutz, C. Areces, I. Horrocks, and U. Sattler. Keys, Nominals, and Concrete Domains. *J. of Artificial Intelligence Research*, 23:667–726, 2004.
13. J. McCarthy. Circumscription – A Form of Non-Monotonic Reasoning. *Artificial Intelligence*, 13(1–2):27–39, 1980.
14. B. Motik, I. Horrocks, R. Rosati, and U. Sattler. Can OWL and Logic Programming Live Together Happily Ever After? In *Proc. of the 5th Int. Semantic Web Conf. (ISWC'06)*, volume 4273 of *LNCS*, pages 501–514. Springer, November 2006.
15. I. Niemelä. Implementing Circumscription Using a Tableau Method. In *Proc. of the 12th Europ. Conf. on Artificial Intelligence (ECAI'96)*. J. Wiley & Sons, 1996.
16. N. Olivetti. A tableaux and Sequent Calculus for Minimal Entailment. *Journal of Automated Reasoning*, 9:99–139, 1992.
17. P. F. Patel-Schneider, P. Hayes, I. Horrocks, and F. van Harmelen. OWL Web Ontology Language; Semantics and Abstract Syntax, W3C Candidate Recommendation. http://www.w3.org/TR/owl-semantics/, November 2002.
18. A. Rector. Defaults, Context, and Knowledge: Alternatives for OWL-Indexed Knowledge Bases. In *Pacific Symposium on Biocomputing*, pages 226–237, 2004.
19. A. Rector et al. OWL Pizzas: Common Errors & Common Patterns from Practical Experience of Teaching OWL-DL. In *Proc. of the 11th Intern. Conf. on World Wide Web*, pages 89–98, 2002.
20. Y. Katz V. Kolovski, B. Parsia. Implementing OWL Defaults. In *CEUR Proceedings of the OWL Experiences and Directions Workshop*, Athens, USA, 2006.