

Intelligente Systeme im WWW

Dr. Anupriya Ankolekar, Dr. Pascal Hitzler, Dr. York Sure, Markus Krötzsch
Sommersemester 2006

Übung 3: Beweisverfahren und OWL (26.6.2006)

Aufgabe 3.1 Die modelltheoretische Semantik von Beschreibungslogiken, und damit auch von OWL DL, kann direkt auf die Semantik der Prädikatenlogik zurückgeführt werden. Betrachten Sie daher noch einmal die Aussagen aus Aufgabe 2.11 und vergegenwärtigen Sie sich deren Bedeutung für Wissensbasen in OWL DL.

Aufgabe 3.2 Übersetzen Sie die folgenden Teile einer OWL/RDF-Datei in die OWL-DL-Syntax und in die OWL-Abstract-Syntax.

- (a) `<owl:ObjectProperty rdf:about="hatBelag">
 <rdfs:subPropertyOf rdf:resource="hatZutat"/>
 <rdfs:range rdf:resource="PizzaBelag"/>
 <rdfs:domain rdf:resource="Pizza"/>
</owl:ObjectProperty>`
- (b) `<owl:Class rdf:about="Gemüse">
 <owl:disjointWith rdf:resource="Käse"/>
 <owl:disjointWith rdf:resource="Fleisch"/>
 <owl:disjointWith rdf:resource="Fisch"/>
</owl:Class>`
- (c) `<owl:Class rdf:ID="KäsePizza">
 <owl:equivalentClass>
 <owl:Restriction>
 <owl:someValuesFrom>
 <owl:Class rdf:about="Käse"/>
 </owl:someValuesFrom>
 <owl:onProperty>
 <owl:ObjectProperty rdf:about="hatBelag"/>
 </owl:onProperty>
 </owl:Restriction>
 </owl:equivalentClass>
 <rdfs:subClassOf rdf:resource="Pizza"/>
</owl:Class>`

Aufgabe 3.3 Es soll das Konzept „vegetarische Pizza“ definiert werden. Welche der folgenden Definitionen ist dafür angemessen? Geben Sie dazu jeweils eine natürlichsprachliche Beschreibung der logischen Formeln an.

- (a) $\text{VegetarischePizza} \equiv \text{Pizza} \sqcap \neg \exists \text{ hatZutat.}(\text{Fleisch} \sqcap \text{Fisch})$
- (b) $\text{VegetarischePizza} \equiv \text{Pizza} \sqcap \forall \text{ hatBelag.}(\neg \text{Fleisch} \sqcup \neg \text{Fisch})$
- (c) $\text{VegetarischePizza} \equiv \text{Pizza} \sqcap \neg \exists \text{ hatBelag.} \text{Fleisch} \sqcap \neg \exists \text{ hatBelag.} \text{Fisch}$
- (d) $\text{VegetarischePizza} \equiv \text{Pizza} \sqcap \exists \text{ hatBelag.} \neg \text{Fleisch} \sqcap \exists \text{ hatBelag.} \neg \text{Fisch}$
- (e) $\text{VegetarischePizza} \equiv \text{Pizza} \sqcap \forall \text{ hatZutat.}(\neg \text{Fleisch} \sqcap \neg \text{Fisch})$

Aufgabe 3.4 Betrachten Sie zusätzlich zu den in Aufgabe 3.2 vorkommenden Aussagen die folgenden Klassendefinitionen:

$$\text{PizzaSpinat} \quad \equiv \exists \text{ hatBelag.} \text{Spinat} \sqcap \exists \text{ hatBelag.} \text{Käse} \sqcap \forall \text{ hatBelag.}(\text{Spinat} \sqcup \text{Käse})$$

$$\text{PizzaCarnivorus} \equiv \text{Pizza} \sqcap \forall \text{ hatBelag.}(\text{Fleisch} \sqcap \text{Fisch})$$

$$\text{LeerePizza} \quad \equiv \text{Pizza} \sqcap \neg \exists \text{ hatBelag.} \top$$

- (a) Welche der oben und in Aufgabe 3.2 aufgeführten Klassen von Pizzas würde durch einen DL-Reasoner als Unterklasse von *VegetarischePizza* (gemäß einer *korrekten* Definition aus der vorigen Aufgabe) erkannt? Begründen Sie jeweils Ihre Entscheidung.
- (b) Die Klassifikation unter (a) zeigt, dass einige der Pizzaklassen nicht das gewünschte Konzept modellieren. Wie könnte man ihre Definition korrigieren?
- (c) Wie würde sich das unter (a) ermittelte Ergebnis verändern, wenn man bei der Definition von *VegetarischePizza* anstelle von \equiv nur \sqsubseteq verwenden würde?

Hinweis zur nächsten Aufgabe: Mit Hilfe von Erweiterungen wie *blocking* wird sichergestellt, dass jedes Tableau für (geeignete) Beschreibungslogiken irgendwann fertiggestellt werden kann. Dies ist dann erreicht, wenn keine Regel mehr sinnvoll anwendbar ist (entweder, weil die entsprechenden Formeln schon früher erzeugt wurden, oder weil die Regelanwendung geblockt ist).

Nach wie vor ist die Herleitung eines *abgeschlossenen* Tableaus ein Beweis für die Unerfüllbarkeit einer Formel (oder Wissensbasis). Wird allerdings das Tableau fertiggestellt und ist dennoch nicht abgeschlossen, dann beweist das die Erfüllbarkeit der Formel. Im Gegensatz zu FOL kann man mit Tableaus in DL also nicht nur Unerfüllbarkeit (und damit Allgemeingültigkeit), sondern auch Erfüllbarkeit (und damit Widerlegbarkeit) nachweisen.

Aufgabe 3.5 Beweisen Sie mit Hilfe des Tableauverfahrens die Erfüllbarkeit oder Unerfüllbarkeit der folgenden Wissensbasen. Verwenden Sie dazu die Regeln von Folie 60. Bedenken Sie, dass die Formeln dazu zuerst in Negationsnormalform umgeformt werden müssen (siehe Folie 53).

- | | | |
|-----|--|--|
| (a) | $Pizza \sqcap PizzaBelag \sqsubseteq \perp$ $\exists hatBelag.PizzaBelag \sqsubseteq Pizza$ $PizzaBelag(käse)$ $PizzaBelag(aubergine)$ $hatBelag(aubergine, käse)$ | <p>Nichts ist gleichzeitig Pizza und Pizzabelag. Alles was einen Pizzabelag hat, ist eine Pizza. Der Käse ist ein Pizzabelag. Die Aubergine ist ein Pizzabelag. Die Aubergine wurde mit Käse belegt.</p> |
| (b) | $Student \sqsubseteq \exists besucht.Vorlesung$ $Vorlesung \sqsubseteq \exists besuchtVon.(Student \sqcap Fleißig)$ $Student(heiner)$ $\neg Fleißig(heiner)$ | <p>Jeder Student besucht eine Vorlesung. In jeder Vorlesung ist auch ein fleißiger Student. Heiner ist Student, aber nicht fleißig.</p> |

Aufgabe 3.6 Wandeln Sie die beiden Wissensbasen aus Aufgabe 3.5 in Klauselform um. Drücken Sie dazu die Formeln zunächst in FOL aus und wandeln Sie diese anschließend in Klauseln um.

Aufgabe 3.7 Der Nutzen des Tableauverfahrens basiert auf der Beobachtung, dass viele typische Fragestellungen an einen beschreibungslogische Wissensbasis auf die (globale) Unerfüllbarkeit einer Wissensbasis zurückgeführt werden können (siehe Folie 49). Ist es umgekehrt auch möglich, globale Unerfüllbarkeit auf die anderen Inferenzprobleme von Folie 45 zurückzuführen? In welchen Fällen geht das und wie?

Aufgabe 3.8 Wie auf Folie 31 dargestellt, unterscheiden sich die verschiedenen Formalismen zur Wissensrepräsentation auch hinsichtlich ihrer Komplexität und Entscheidbarkeit. Entscheiden Sie, welche der folgenden Erläuterungen der Aussagen von Folie 31 korrekt sind!

- (a) OWL Full ist unentscheidbar. Es gibt also mindestens eine OWL-Full-Wissensbasis, für die kein Algorithmus entscheiden kann, ob sie erfüllbar ist.
- (b) OWL DL ist NEXPTIME-vollständig. Das heißt, auch ein Algorithmus, der nichtdeterministisch immer die richtigen Entscheidungen trifft, benötigt in manchen Fällen exponentiell viel Zeit zur Beantwortung einer Frage.
- (c) OWL Lite ist EXPTIME-vollständig. Das heißt, logisches Schließen in OWL Lite ist schon aus theoretischer Sicht sehr viel schwieriger als das Lösen von NP-vollständigen Problemen.¹

¹Ein Beispiel für ein NP-vollständiges Problem ist das logische Schließen in der Aussagenlogik.