Prof. Dr. Thomas Schneider, Dr. Jean Christoph Jung SoSe 2020

Beschreibungslogik

Übungsblatt 3

Abgabe im PDF-Format bis 3.6.2020, 23:59 Uhr in Stud.IP, Ordner „Abgabe Blatt 3“
Bitte nur eine PDF-Datei pro Gruppe, Lizenz „Selbst verfasstes, nicht publiziertes Werk“.

---

1. (20 %) Verwende den Tableau-Algorithmus für $\mathcal{ALC}$ aus der Vorlesung, um die Erfüllbarkeit der folgenden Konzepte zu entscheiden.

a) $C_0 = \exists r.A \sqcap \exists r.B \sqcap \forall r.\exists r.A \sqcap \forall r.\exists r.\neg A \sqcap \forall r.\forall r.\neg A$

b) $C_0 = \neg\big(\forall r.(A \sqcup B) \sqcap \forall r.(A \sqcup \neg B)\big) \sqcap \neg\exists r.(\neg A \sqcap \neg B)$

Gib an, welche Regeln in welcher Reihenfolge worauf angewendet werden (orientiere Dich am Beispiel aus der Vorlesung). Gib im Falle von Erfüllbarkeit außerdem ein Modell gemäß dem Beweis von Theorem 4.8 an.

2. (20 %) Im Computerspiel *Multimonster Massacre* bist Du in einem Raum mit $n \geq 0$ Monstern gefangen. Jedes Monster hat bis zu 100 Köpfe. Du hast eine unbegrenzte Anzahl von Schlägen zur Verfügung. Mit jedem Schlag erlegst Du genau ein Monster, welches dann verschwindet und sofort durch eine beliebige Anzahl neuer Monster ersetzt wird, von denen aber jedes strikt weniger Köpfe hat als das erlegte. Ein kopfloses Monster verwandelt sich sofort in einen Keks.

Zeige mittels einer Multimengenordnung, dass nach endlich vielen Schlägen nur noch Kekse im Raum sind.

3. (20 %) Verwende den Tableau-Algorithmus für $\mathcal{ALC}$ mit TBoxen aus der Vorlesung um zu entscheiden, ob

a) $C_1 = \exists r.(\exists r.A \sqcup \exists r.B)$ erfüllbar bzgl. $\mathcal{T}_1 = \{\top \sqsubseteq \forall r.(\neg A \sqcap \neg B)\}$ ist;

b) $C_2 = \neg A \sqcap B \sqcap \forall r.(B' \sqcap \forall r.\neg A)$ erfüllbar bzgl. $\mathcal{T}_2 = \{\top \sqsubseteq \exists r.B \sqcap \exists s.B\}$ ist;

c) die Subsumtion $\mathcal{T}_3 \models \mathsf{Student} \sqsubseteq \mathsf{Happy}$ gilt,
wobei $\mathcal{T}_3 = \{\mathsf{Student} \sqsubseteq \exists \mathsf{solves}.\mathsf{Exercise},\ \exists \mathsf{solves}.\top \sqsubseteq \mathsf{Happy}\}$.

4. (20 %) Benutze Protégé 5.5 und einen Reasoner (z. B. `HermiT`, `FaCT++` oder `Pellet`), um Deine Ergebnisse aus Aufgabe 3 zu überprüfen. Schreibe dazu jeweils eine OWL-Ontologie mit den Konzeptinklusionen aus $\mathcal{T}_i$ und nutze bei Bedarf die unten gegebenen Hinweise sowie die in Aufgabe 5 b) auf Blatt 1 genannten Hilfsmittel.

Lege Deiner Abgabe die drei Ontologien und je einen Screenshot der *Class hierarchy (Inferred)* nach dem Klassifizieren bei (bitte alles in eine ZIP-Datei packen).

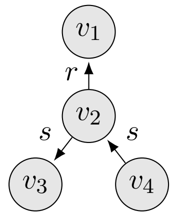
Hinweise: (1) Um in Protégé zu prüfen, ob ein komplexes Konzept C erfüllbar bezüglich einer TBox $\mathcal{T}$ ist, füge der OWL-Ontologie, die Du für $\mathcal{T}$ geschrieben hast, einen neuen Konzeptnamen X und die Definition $X \equiv C$ hinzu und klassifiziere die Ontologie. Wenn der Reasoner meldet, dass sie inkonsistent ist, dann hat $\mathcal{T}$ keine Modelle, und C ist somit *un*erfüllbar bzgl. $\mathcal{T}$. Ist sie konsistent, dann öffne nach dem Klassifizieren die Ansicht *Class hierarchy*, wechsle dort zu *Inferred* und schaue nach, ob X rot gefärbt ist und unter `owl:Nothing` erscheint (genau dann ist X *un*erfüllbar bzgl. $\mathcal{T}$).

(Wenn C ein Konzeptname ist, kannst Du auf die Definition $X \equiv C$ verzichten.)

(2) Um zu prüfen, ob eine Subsumtion $\top \sqsubseteq C$ bzgl. einer TBox $\mathcal{T}$ gilt, füge wie in (1) wieder eine neue Definition $X \equiv C$ hinzu und überprüfe, ob nach dem Klassifizieren X neben `owl:Thing` erscheint.

Bitte wenden.

5. (20 %) Betrachte die folgende Erweiterung des Tableau-Algorithmus mit TBoxen aus der Vorlesung auf die Beschreibungslogik $\mathcal{ALCI}$:



1. Erweiterung von I-Bäumen:
   - Kanten dürfen vorwärts und rückwärts zeigen; im nebenstehenden Beispiel ist $E = \{(v_2, r, v_1),\ (v_2, s, v_3),\ (v_4, s, v_2)\}$.
2. zusätzliche $\exists^-$-Regel:
   - Wähle $v \in V$ und $\exists r^-.C \in \mathcal{L}(v)$, so dass v nicht blockiert ist und es kein $v' \in V$ gibt mit $\boldsymbol{(v', r, v) \in E}$ und $C \in \mathcal{L}(v')$.
   - Erweitere V um neuen Knoten v' und E um $\boldsymbol{(v', r, v)}$, und setze $\mathcal{L}(v') = \{C\}$.
3. zusätzliche $\forall^-$-Regel:
   - Wähle $v, v' \in V$ und $\forall r^-.C \in \mathcal{L}(v)$, so dass $\boldsymbol{(v', r, v) \in E}$ und $C \notin \mathcal{L}(v')$.
   - Erweitere $\mathcal{L}(v')$ um C.

Zeige, dass der skizzierte Algorithmus auf der folgenden Eingabe ein falsches Ergebnis liefert: $C_0 = A$ und $\mathcal{T} = \{\top \sqsubseteq \exists r.A,\ \top \sqsubseteq \forall r^-.\forall r^-.\neg A\}$

6. Zusatzaufgabe (20 %) Mit $\mathcal{ALC}_{\mathsf{trans}}$ bezeichnen wir die Erweiterung von $\mathcal{ALC}$ um transitive Rollen, d. h. die TBox darf nun zusätzlich zu Konzeptinklusionen auch Zusicherungen der Form $\mathsf{trans}(r)$ enthalten, wobei r ein Rollenname ist. Eine Interpretation $\mathcal{I}$ erfüllt $\mathsf{trans}(r)$, wenn $r^{\mathcal{I}}$ eine transitive Relation ist.

Sei $\mathcal{T}$ eine $\mathcal{ALC}_{\mathsf{trans}}$-TBox der Form $\{\top \sqsubseteq C_{\mathcal{T}}\}$ mit $C_{\mathcal{T}}$ in NNF. Wir definieren eine $\mathcal{ALC}$ TBox $\mathcal{T}^*$ wie folgt:

- $\mathcal{T}^*$ enthält $\top \sqsubseteq C_{\mathcal{T}}$.
- Für jedes $\forall r.C \in \mathrm{sub}(C_{\mathcal{T}})$ mit $\mathsf{trans}(r) \in \mathcal{T}$ enthält $\mathcal{T}^*$ die Konzeptinklusion $\forall r.C \sqsubseteq \forall r.\forall r.C$.

Beweise, dass für alle Konzeptnamen A gilt:

$$A \text{ ist erfüllbar bzgl. } \mathcal{T} \quad \text{gdw.} \quad A \text{ ist erfüllbar bzgl. } \mathcal{T}^* \qquad (*)$$

Hinweise:

Für die Richtung "⇒" zeige, dass jedes Modell von A und $\mathcal{T}$ auch ein Modell von $\mathcal{T}^*$ ist (das ist die einfache Richtung).

Für die Richtung "⇐" beschreibe, wie man aus einem Modell $\mathcal{I}$ von A und $\mathcal{T}^*$ ein Modell $\mathcal{J}$ von $\mathcal{T}$ konstruieren kann. Um zu zeigen, dass $\mathcal{J}$ Modell von $\mathcal{T}$ ist, beweise folgende Hilfsaussage:

$$\text{Für alle } C \in \mathsf{sub}(C_{\mathcal{T}}) \text{ und alle } d \in \Delta^{\mathcal{I}} \text{ gilt: } d \in C^{\mathcal{I}} \text{ impliziert } d \in C^{\mathcal{J}}.$$

Die bewiesene Aussage $(*)$ besagt also, dass es sich hier um eine Reduktion handelt, die es erlaubt, den Tableau-Algorithmus für $\mathcal{ALC}$ auch für $\mathcal{ALC}_{\mathsf{trans}}$ zu verwenden.