Humboldt-Universität zu Berlin
Lehrstuhl Logik in der Informatik
Prof. Dr. Nicole Schweikardt

6. Januar 2023

Logik in der Informatik

Wintersemester 2022/2023

Übungsblatt 10

Abgabe: *bis 16. Januar 2023, 13.^{00} Uhr*

Aufgabe 1: (Moodle-Quiz)

Absolvieren Sie das Quiz 10 auf der Moodle-Plattform.

Aufgabe 2: (Präsenzaufgabe)

(a) Sei σ eine beliebige Signatur. Zeigen Sie, dass es Formeln $\varphi, \psi \in \mathsf{FO}[\sigma]$ gibt, für die gilt:

$$(\,\varphi \vee \exists x \psi\,) \;\not\equiv\; \exists x (\,\varphi \vee \psi\,)$$

(b) Hunde äußern sich bekanntlich mit Hilfe der Laute „W“, „A“ und „U“.
Sei $\Sigma := \{\mathrm{W}, \mathrm{A}, \mathrm{U}\}$ und sei die *Hundesprache* H definiert durch $H := \mathrm{abl}_{\mathfrak{K}}$, wobei $\mathfrak{K}$ der folgende Kalkül über der Menge Σ^* ist:

$$\begin{aligned}
\mathfrak{K} \;:=\quad & \left\{ \frac{}{\mathrm{WA}} \right\} \\
\cup\quad & \left\{ \frac{v}{vv} : \text{ für alle } v \in \Sigma^* \right\} \\
\cup\quad & \left\{ \frac{v\mathrm{A}w}{v\mathrm{U}w} : \text{ für alle } v, w \in \Sigma^* \right\} \\
\cup\quad & \left\{ \frac{v\mathrm{UU}w}{v\mathrm{AAA}w} : \text{ für alle } v, w \in \Sigma^* \right\} \\
\cup\quad & \left\{ \frac{v\mathrm{AAA}w}{vw} : \text{ für alle } v, w \in \Sigma^* \right\}
\end{aligned}$$

(i) Geben Sie für jedes der folgenden Worte aus Σ^* an, ob es zur Menge H gehört oder nicht. Begründen Sie jeweils Ihre Antwort!

(i) WA **(ii)** UWAA **(iii)** WAWAUU **(iv)** WU

(ii) Zeigen Sie, dass für jedes Wort $w \in H$ gilt:
Die Anzahl $|w|_A$ der Vorkommen des Lauts A in w ist *nicht* durch 3 teilbar
(d.h., es gibt eine Zahl $k \in \mathbb{N}$, so dass gilt: $|w|_A = 3k + 1$ oder $|w|_A = 3k + 2$).

(iii) Kann ein Hund „WAAA“ machen? D.h., ist WAAA $\in H$?

Aufgabe 3: (40 Punkte)

(a) Welche der folgenden beiden Aussagen ist für jede Signatur σ und alle $\mathsf{FO}[\sigma]$-Formeln φ und ψ korrekt, welche nicht? Beweisen Sie, dass ihre Antworten korrekt sind.

(i) $\exists x \forall y \varphi \models \forall y \exists x \varphi$ **(ii)** $\forall y \exists x \varphi \models \exists x \forall y \varphi$

(b) Sei $\sigma = \{E\}$ die Signatur mit dem 2-stelligen Relationssymbol E. Betrachten Sie die $\mathsf{FO}[\sigma]$-Formel

$$\varphi(x) \quad := \quad \neg \exists y \forall z \Big(E(y,z) \to (E(x,z) \wedge \exists x\, E(x,x)) \Big)$$

(i) Berechnen Sie eine zu φ äquivalente $\mathsf{FO}[\sigma]$-Formel in Negationsnormalform.

(ii) Berechnen Sie eine zu φ äquivalente $\mathsf{FO}[\sigma]$-Formel in Pränex-Normalform.

Gehen Sie hierbei ähnlich wie in Beispiel 3.70 vor. Machen Sie pro Zwischenschritt nur eine Umformung und kommentieren Sie Ihre Zwischenschritte.

(c) Beweisen Sie Satz 3.67 aus der Vorlesung, das heißt zeigen Sie:

Jede $\mathsf{FO}[\sigma]$-Formel φ ist äquivalent zu einer Formel in NNF.

(d) Sei $\Sigma = \{a, b\}$ und $\sigma_\Sigma = \{\leqslant, P_a, P_b\}$ die Signatur, die aus dem 2-stelligen Relationssymbol $\leqslant$, sowie zwei 1-stelligen Relationssymbolen P_a und P_b besteht. Beweisen Sie mittels **logischer Reduktion**, dass es keinen $\mathsf{FO}[\sigma_\Sigma]$-Satz gibt, der die Sprache aller nicht-leeren Worte aus Σ^* beschreibt, in denen die Anzahl der in ihnen vorkommenden as gleich der Anzahl der in ihnen vorkommenden bs ist.

Zur Erinnerung: Ein $\mathsf{FO}[\sigma_\Sigma]$-Satz φ *beschreibt* eine Sprache $L \subseteq \Sigma^*$, falls für jedes nicht-leere Wort $w \in \Sigma^*$ gilt: $w \in L \iff \mathcal{A}_w \models \varphi$.

Aufgabe 4: (20 Punkte)

Lesen Sie Kapitel 11 aus dem Buch „Learn Prolog Now!“.
Achtung: *Die Bearbeitung der Aufgabe ist unter Beachtung der bekannten Abgabehinweise über Moodle abzugeben! Importieren Sie in Ihrer Abgabedatei* `blatt10.pl`*, analog zu Blatt 7, die Dateien* `al.pl` *und* `kinodb.pl`*.*

(a) Schreiben Sie ein Prädikat `clooney/1`, so dass die Anfrage

```
?- clooney(R).
```

in der Liste `R` die Menge aller Tupel `(K, Z)` zurückgibt, so dass gilt: Zum Zeitpunkt `Z` läuft im Kino `K` ein Film, in dem George Clooney mitgespielt hat.

(b) Wir implementieren eine Reservierungsverwaltung für das Kino Babylon (in dem zur Zeit aus technischen Gründen nur ein Saal in Betrieb ist). Der Umstand, dass eine Person `P` für den Film, der um `Z` Uhr beginnt, Sitzplatz `S` reserviert hat, soll durch einen Fakt `belegt(P, Z, S)` in der Wissensbasis ausgedrückt werden. Stellen Sie zu diesem Zweck Ihrer Datei `blatt10.pl` die Zeile

```
:- dynamic belegt/3.
```

voran. Schreiben Sie ein Prädikat `reservieren/3`, so dass die Anfrage

```
?- reservieren(P, Z, S).
```

den Sitzplatz `S` für Person `P` und `Z` Uhr reserviert, d.h., der Wissensbasis einen Fakt `belegt(P, Z, S)` hinzufügt. Dies soll allerdings nur möglich sein, wenn um `Z` Uhr im Babylon tatsächlich ein Film läuft, und wenn der Sitzplatz für diese Uhrzeit noch nicht belegt ist. Anderenfalls soll die Anfrage scheitern.

(c) Schreiben Sie ein Prädikat `stornieren/2`, so dass die Anfrage

```
?- stornieren(Z, S).
```

die Reservierung für den Sitzplatz `S` zur Zeit `Z` aufhebt, d.h., einen entsprechenden Fakt in der Wissensbasis löscht. Gibt es eine solche Reservierung nicht, so soll die Anfrage scheitern.

(d) Wir repräsentieren im Folgenden Klauseln als Listen von Literalen und Klauselmengen als Listen von Klauseln. So repräsentiert `[[~x1, ~x2, ~x3, x4], [x1, ~x2], [x2]]` die Klauselmenge $\{\{\neg X_1, \neg X_2, \neg X_3, X_4\}, \{X_1, \neg X_2\}, \{X_2\}\}$. Schreiben Sie ein Prädikat `unit_propagation/2`, das die Vereinfachungsheuristik *Unit Propagation* des DPLL-Algorithmus implementiert. D.h., ist `K` eine Klauselmenge, dann sollte die Anfrage

```
?- unit_propagation(K, K2).
```

in `K2` die Klauselmenge zurückgeben, die aus `K` entsteht, indem die Unit Propagation so lange auf `K` angewendet wird, bis keine „Einerklausel“ mehr vorhanden ist. Beispielsweise sollte die Anfrage

```
?- unit_propagation([[~x1, ~x2, ~x3, x4], [x1, ~x2], [x2]], K2).
```

zu der folgenden (oder einer äquivalenten) Antwort führen:

```
K2 = [[~x3, x4]].
```

Hinweis: Führen Sie geeignete Hilfsprädikate ein.