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

21. Januar 2022

Logik in der Informatik

Wintersemester 2021/2022

Übungsblatt 12

Abgabe: *bis 31. Januar 2022, 13.⁰⁰ Uhr*

Aufgabe 1: (Moodle-Quiz)

Absolvieren Sie das Quiz 12 auf der Moodle-Plattform.

Aufgabe 2: (Präsenzaufgabe)

(a) Sei $\sigma := \{R, f_0, f_1, c\}$, wobei c ein Konstantensymbol, R ein 2-stelliges Relationssymbol und f_0, f_1 zwei 1-stellige Funktionssymbole sind.

Beweisen Sie folgende Aussagen aus Korollar 4.41 aus dem Vorlesungsskript:

- **(i)** Das Unerfüllbarkeitsproblem für $\mathsf{FO}[\sigma]$ ist nicht entscheidbar.
- **(ii)** Das Erfüllbarkeitsproblem für $\mathsf{FO}[\sigma]$ ist nicht semi-entscheidbar.

(b) Sei R ein 2-stelliges Relationssymbol und seien c und d Konstantensymbole.

Sei $\sigma := \{R, c, d\}$ und sei φ der folgende $\mathsf{FO}[\sigma]$-Satz:

$$\Big(\exists x \exists y \, \big(\, R(x,d) \;\wedge\; R(c,y) \big) \;\;\wedge\;\; \forall x \forall y \, \big(\, R(x,y) \;\rightarrow\; \neg\, x{=}y \, \big)\Big)$$

Geben Sie eine σ-Herbrandstruktur $\mathcal{A}$ und eine σ-Herbrandstruktur $\mathcal{B}$ an, so dass gilt:

$$\mathcal{A} \models \varphi \qquad \text{und} \qquad \mathcal{B} \not\models \varphi.$$

Begründen Sie jeweils, warum $\mathcal{A} \models \varphi$ bzw. $\mathcal{B} \not\models \varphi$ gilt.

Aufgabe 3: **(40 Punkte)**

(a) Sei σ eine Signatur, sei $\Gamma \subseteq_e \mathsf{FO}[\sigma]$, seien $\varphi, \psi \in \mathsf{FO}[\sigma]$, $t \in \mathsf{T}_\sigma$ und seien $x, y \in \mathsf{VAR}$.

(i) Beweisen Sie die Korrektheit der Sequenzenregel $\exists$-Einführung im Antezedens ($\exists$A):

$$\frac{\Gamma, \varphi\frac{y}{x} \ \vdash \psi}{\Gamma, \exists x\varphi \vdash \psi} \qquad \text{falls} \ \ y \notin \text{frei}(\Gamma,\ \exists x\varphi,\ \psi)$$

(ii) **Beweisen oder widerlegen** Sie die Korrektheit der folgenden Sequenzenregel

$$\frac{\begin{array}{ll}\Gamma, \psi \vdash & \varphi \\ \Gamma, \varphi \vdash & \psi\end{array}}{\begin{array}{ll}\Gamma & \vdash (\varphi \vee \psi)\end{array}}$$

(iii) **Beweisen oder widerlegen** Sie die Korrektheit der folgenden Sequenzenregel

$$\frac{\begin{array}{ll}\Gamma, \varphi \vdash & \psi\end{array}}{\begin{array}{ll}\Gamma & \vdash (\neg\varphi \vee \psi)\end{array}}$$

(b) Sei σ eine Signatur und seien $\varphi, \psi \in \mathsf{FO}[\sigma]$.
Geben Sie für die folgende Sequenz eine Ableitung im Sequenzenkalkül $\mathfrak{K}_S$ an oder beweisen Sie, dass eine solche Ableitung nicht existiert

$$\{\ \varphi,\ (\neg\varphi \vee \psi)\ \} \ \vdash \ \psi$$

(c) Sei $\sigma := \{E\}$ die Signatur, die aus einem 2-stelligen Relationssymbol E besteht.

(i) Zeigen Sie, dass die Klasse aller azyklischen (endlichen oder unendlichen) Graphen erststufig axiomatisierbar ist.

(ii) Nutzen Sie den Endlichkeitssatz der Logik erster Stufe um zu zeigen, dass die Klasse aller *nicht* azyklischen (endlichen oder unendlichen) Graphen *nicht* erststufig axiomatisierbar ist.

Zur Erinnerung: Ein gerichteter Graph ist azyklisch, falls er keinen Kreis endlicher Länge besitzt.

— auf der nächsten Seite geht's weiter —

Aufgabe 4: (20 Punkte)

Lesen Sie Kapitel 7 aus dem Buch „Learn Prolog Now!“.
Achtung: *Die Bearbeitung der Aufgabe ist unter Beachtung der bekannten Abgabehinweise über Moodle abzugeben! Analog zu früheren Blättern finden Sie die benötigten Dateien auf der Seite zur Prolog-Übung. Machen Sie sich auch mit den neuen Prolog-Modulen* ***`unit_propagation`*** *und* ***`pure_literal`*** *vertraut.*[1]

(a) Importieren Sie im Modul `dpll` Ihrer Abgabe `blatt12.pl` die Prädikate aus den Prolog-Modulen, die Sie für die Lösung der folgenden Teilaufgabe benötigen.

(b) Wir kodieren Klauselmengen, wie gewohnt, als Listen von Listen von Literalen. Implementieren Sie das Prädikat `dpll/1`, so dass eine Anfrage

```
?- dpll(KM).
```

für eine Klauselmenge `KM` genau dann erfolgreich ist, wenn die Klauselmenge erfüllbar ist.

Beispielsweise sollte die Anfrage für die Klauselmenge

```
KM = [[x1,~x5,~x6,x7], [~x1,x2,~x5], [~x1,~x2,~x3,~x5,~x6],
      [x1,x2,~x4,x7], [~x4,~x6,~x7], [x3,~x5,x7], [x3,~x4,~x5],
      [x5,~x6], [x5,x4,~x8], [x1,x3,x5,x6,x7], [~x7,x8],
      [~x6,~x7,~x8]]
```

erfolgreich sein. Es macht hierbei nichts, wenn die Antwort `true.` durch das Backtracking mehrfach ausgegeben werden kann. Für die Klauselmenge

```
KM = [[~r,t,w], [~r,~s,~w], [~r,~t], [~q,s,t], [~q,r,~s],
      [r,s,w], [r,~t,~w], [q,u], [s,~u,~w], [q,w], [q,~s,~u]]
```

sollte die selbe Anfrage jedoch scheitern.
Hinweise: Implementieren Sie dazu den *DPLL-Algorithmus*, wie er auf Seite 89 des Skripts beschrieben ist. Definieren Sie geeignete Hilfsprädikate. Nutzen Sie insbesondere die bereits auf Blatt 9 und 10 (bzw. 11) implementierten Vereinfachungsheuristiken *Unit Propagation* und *Pure Literal Rule*, die Sie aus den Modulen des entsprechenden Namens importieren können. Die Streichung von Klauseln, die Obermengen von anderen Klauseln sind, müssen Sie nicht implementieren.

(c) Gegeben sei die kontextfreie Grammatik $G = (\Sigma, V, S, P)$ mit den Terminalsymbolen $\Sigma := \{\texttt{if}, \texttt{then}, \texttt{else}, \texttt{e1}, \texttt{e2}, \texttt{s1}, \texttt{s2}\}$, den Nichtterminalsymbolen $V := \{\texttt{stmt}, \texttt{expr}\}$, dem Startsymbol $S := \texttt{stmt}$, und den Produktionen $P :=$

$$\{\ \texttt{stmt} \longrightarrow \texttt{if expr then stmt},\quad \texttt{stmt} \longrightarrow \texttt{if expr then stmt else stmt},\\ \texttt{stmt} \longrightarrow \texttt{s1},\quad \texttt{stmt} \longrightarrow \texttt{s2},\quad \texttt{expr} \longrightarrow \texttt{e1},\quad \texttt{expr} \longrightarrow \texttt{e2}\ \}$$

Bilden Sie für die kontextfreie Grammatik G eine *Definite Clause Grammar (DCG)*, so dass die Anfrage

```
?- stmt(X, []).
```

genau dann erfüllt wird, wenn `X` eine Liste von Terminalsymbolen aus Σ ist, die einem Wort der durch G beschriebenen Sprache entspricht. Dies gilt beispielsweise für die Liste

```
X = [ if, e1, then, if, e2, then, s1, else, s2] .
```

Fügen Sie Ihre Definite Clause Grammar der Datei `blatt12.pl` hinzu.

[1] Verfügbar spätestens ab 24.01.22 20:00 Uhr.