

Logik in der Informatik

Wintersemester 2020/2021

Übungsblatt 11

Abgabe: bis 8. Februar 2021, 13.00 Uhr via Moodle

Um unseren Tutor*innen die Arbeit etwas zu erleichtern, notieren Sie bitte auf dem ersten Blatt jeder schriftlich erstellten Abgabedatei oben rechts eingerahmt folgende Daten:

<Vorname Nachname>, <Matrikelnr.>, <Moodle-Loginname>

Aufgabe 1:

(Moodle-Quiz)

Absolvieren Sie das Quiz 11 auf der Moodle-Plattform.

Aufgabe 2:

(Präsenzaufgabe)

Wir betrachten in dieser Aufgabe Kalküle über der Menge $M := \text{AL}(\{\neg, \rightarrow\})$.

Ein Kalkül \mathfrak{K} über M heißt *korrekt*, falls für jede Menge $\Phi \subseteq M$ und jede Formel $\psi \in M$ gilt: Wenn $\psi \in \text{abl}_{\mathfrak{K}}(\Phi)$, dann gilt $\Phi \models \psi$. Ein Kalkül \mathfrak{K} über M heißt *vollständig*, falls für jede Menge $\Phi \subseteq M$ und jede Formel $\psi \in M$ gilt: Wenn $\Phi \models \psi$, dann ist $\psi \in \text{abl}_{\mathfrak{K}}(\Phi)$.

Seien \mathfrak{K}_{Syl} und \mathfrak{K}_{Abd} die beiden folgenden Kalküle über der Menge M : Beide Kalküle enthalten für jede *allgemeingültige* aussagenlogische Formel $\varphi \in M$ das Axiom $\frac{}{\varphi}$.

Außerdem enthält

- \mathfrak{K}_{Abd} für alle $\varphi, \psi \in M$ die Ableitungsregel

$$\frac{\psi \quad (\varphi \rightarrow \psi)}{\varphi},$$

die *Abduktion* genannt wird,

- \mathfrak{K}_{Syl} für alle $\varphi, \psi, \chi \in M$ die Ableitungsregel

$$\frac{(\varphi \rightarrow \psi) \quad (\psi \rightarrow \chi)}{(\varphi \rightarrow \chi)},$$

die *Syllogismus* genannt wird.

- Geben Sie für $\varphi := \neg(A_0 \rightarrow A_0)$ und $\Phi := \emptyset$ eine möglichst kurze Ableitung von φ aus Φ in \mathfrak{K}_{Abd} an.
- Beweisen Sie, dass \mathfrak{K}_{Abd} vollständig, aber nicht korrekt ist.
- Beweisen Sie, dass \mathfrak{K}_{Syl} korrekt, aber nicht vollständig ist.
- Betrachten Sie den Kalkül \mathfrak{K} über M , der alle Ableitungsregeln aus \mathfrak{K}_{Syl} und alle Ableitungsregeln aus \mathfrak{K}_{Abd} enthält. Ist \mathfrak{K} korrekt? Ist \mathfrak{K} vollständig? Begründen Sie jeweils Ihre Antwort.

Aufgabe 3:

(40 Punkte)

- (a) Sei σ eine Signatur, sei $\Gamma \subseteq_e \text{FO}[\sigma]$, seien $\varphi, \psi, \chi \in \text{FO}[\sigma]$ und seien $x, y \in \text{VAR}$.
Beweisen Sie die Korrektheit der folgenden Sequenzenregeln:

- (i) \wedge -Einführung im Sukzedens ($\wedge S$):

$$\frac{\Gamma \vdash \varphi \quad \Gamma \vdash \psi}{\Gamma \vdash (\varphi \wedge \psi)}$$

- (ii) \exists -Einführung im Antezedens ($\exists A$):

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

- (b) 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.

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 `dp11` Ihrer Abgabe `blatt11.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 `dp11/1`, so dass eine Anfrage

?- `dp11(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]]
```

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

¹Verfügbar ab 01.02.21 ca. 20:00 Uhr.

$$\text{KM} = [[\sim r, t, w], [\sim r, \sim s, \sim w], [\sim r, \sim t], [\sim q, s, t], [\sim q, r, \sim s], \\ [r, s, w], [r, \sim t, \sim w], [q, u], [s, \sim u, \sim w], [q, w], [q, \sim s, \sim u]]$$

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 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 $\Sigma := \{\text{if, then, else, e1, e2, s1, s2}\}$, den Nichtterminalsymbolen $V := \{\text{stmt, expr}\}$, dem Startsymbol $S := \text{stmt}$, und den Produktionen $P :=$

$$\left\{ \begin{array}{ll} \text{stmt} \longrightarrow \text{if expr then stmt}, & \text{stmt} \longrightarrow \text{if expr then stmt else stmt}, \\ \text{stmt} \longrightarrow \text{s1}, & \text{stmt} \longrightarrow \text{s2}, \quad \text{expr} \longrightarrow \text{e1}, \quad \text{expr} \longrightarrow \text{e2} \end{array} \right\}$$

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

$$?- \text{stmt}(X, []).$$

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

$$X = [\text{if, e1, then, if, e2, then, s1, else, s2}] .$$

Fügen Sie Ihre Definite Clause Grammar der Datei `blatt11.pl` hinzu.