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

7. Dezember 2017

Logik in der Informatik

Wintersemester 2017/2018

Übungsblatt 8

Abgabe: *bis 19. Dezember 2017, 11.10 Uhr (vor der Vorlesung oder im Briefkasten zwischen den Räumen 3.401 und 3.402 im Johann von Neumann-Haus (Rudower Chaussee 25))*

Aufgabe 1: (25 Punkte)

Betrachten Sie die Kinodatenbank $\mathcal{D}$ aus der Vorlesung (Skript, Kapitel 3.1).

(a) Geben Sie für die folgenden Anfragen jeweils eine $\mathsf{FO}[\sigma_{\mathsf{KINO}}]$-Formel φ und ein Variablentupel $(x_1, \ldots, x_n)$ mit $\text{frei}(\varphi) \subseteq \{x_1, \ldots, x_n\}$ an, die die Anfrage beschreiben. Berechnen Sie jeweils auch die Relation $[\![\varphi(x_1, \ldots, x_n)]\!]^{\mathcal{D}}$.

- **(i)** Geben Sie alle Schauspieler aus, die in mindestens einem Film mitspielen, in dem George Clooney mitspielt.
- **(ii)** Geben Sie alle Kinos und deren Adresse aus, in denen um '20:00' ein Film beginnt.
- **(iii)** Geben Sie die Titel aller Filme aus, die in genau einem Kino laufen.

(b) Geben Sie umgangssprachlich an, welche Anfragen durch die Formeln φ_1, φ_2 und φ_3 beschrieben werden.

- **(i)** $\varphi_1(x) \quad := \quad \exists x_Z \, R_{Prog}(x, \text{'Alien'}, x_Z)$
- **(ii)** $\varphi_2(x_1, x_2) \quad := \quad \exists x_A \exists x_T \left(R_{Kino}(x_1, x_A, x_2, x_T) \ \wedge \ \exists x_R \exists x_S \, R_{Film}(x_1, x_R, x_S) \right)$
- **(iii)** $\varphi_3(x_1, x_2, x_3, x_4) \quad := \quad \Big(\exists x_T \, R_{Kino}(x_1, x_2, x_3, x_T) \wedge \exists x_{Z_1} \big(R_{Prog}(x_1, x_4, x_{Z_1}) \wedge \forall x_K \forall x_{Z_2} (R_{Prog}(x_K, x_4, x_{Z_2}) \ \rightarrow \ x_K = x_1) \big) \Big)$

Aufgabe 2: (25 Punkte)

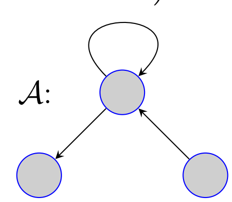
Seien $\sigma := \{E, g, c\}$ und $\sigma' := \{E\}$ Signaturen. Hierbei ist E ein 2-stelliges Relationssymbol, g ein 1-stelliges Funktionssymbol und c ein Konstantensymbol.

(a) Bestimmen Sie für jede der folgenden $\mathsf{FO}[\sigma]$-Formeln φ, welche freie und welche gebundenen Vorkommen von Variablen in φ existieren und geben Sie die Menge $\text{frei}(\varphi)$ aller freien Variablen von φ an (ohne Begründung). Entscheiden Sie außerdem für jede der $\mathsf{FO}[\sigma]$-Formeln, ob es sich um einen $\mathsf{FO}[\sigma]$-Satz handelt.

- **(i)** $g(x) = g(c)$
- **(ii)** $\exists z \left(E(x, z) \ \wedge \ E(z, y) \right)$
- **(iii)** $\forall x \forall y \left(\exists z \, g(z) = x \ \rightarrow \ \exists z \, g(z) = y \right)$
- **(iv)** $\left(\forall z \exists x \, E(x, c) \ \wedge \ \forall y \, E(y, z) \right)$

(b) Betrachten Sie die σ'-Struktur $\mathcal{A} = (A, E^{\mathcal{A}})$, die durch den Graphen in der nebenstehenden Abbildung repräsentiert wird. Geben Sie einen $\mathsf{FO}[\sigma']$-Satz φ an, der die Struktur eindeutig beschreibt. D.h. es soll für alle σ'-Strukturen $\mathcal{B}$ gelten:

$\mathcal{A}$:



$$\mathcal{A} \cong \mathcal{B} \quad \Longleftrightarrow \quad \mathcal{B} \models \varphi$$

(c) Geben Sie für die $\mathsf{FO}[\sigma']$-Formel

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

eine σ'-Struktur $\mathcal{A}$, deren Universum aus höchstens 4 Elementen besteht, und zwei σ'-Interpretationen $\mathcal{I}_1 = (\mathcal{A}, \beta_1)$ und $\mathcal{I}_2 = (\mathcal{A}, \beta_2)$ an, so dass $\mathcal{I}_1 \models \varphi$ und $\mathcal{I}_2 \models \neg\varphi$ gilt.

Aufgabe 3: (25 Punkte)

(a) Beweisen Sie das Koinzidenzlemma für Terme (Satz 3.27) per Induktion über den Aufbau von Termen.

(b) Beweisen Sie das Koinzidenzlemma für Formeln der Logik erster Stufe (Satz 3.28) per Induktion über den Aufbau von Formeln.

Aufgabe 4: (25 Punkte)

Lesen Sie Kapitel 10 aus dem Buch „Learn Prolog Now!“.

Achtung: *Die Bearbeitung der Teilaufgaben* **(b)**, **(c)** *und* **(d)** *ist unter Beachtung der bekannten Abgabehinweise über Moodle abzugeben! Importieren Sie in Ihrer Abgabedatei* `blatt8.pl`*, wie in Blatt 7, die Datei* `al.pl`*.*

(a) Gegeben sei das folgende Prolog-Programm:

```
a(X, Y) :- b(X, Y).
a(1, 1).
b(X, X) :- c(X).
b(X, Y) :- c(X), !, c(Y).
b(X, Y) :- c(X), c(Y).
c(2).
c(3).
```

Zeichnen Sie einen Suchbaum für die folgende Anfrage: `?- a(X, Y).`

(b) Schreiben Sie in `blatt8.pl` ein Prädikat `not_member/2`, so dass `not_member(X, L)` für einen Term `X` und eine Liste `L` genau dann erfüllt ist, wenn `X` mit *keinem* Element von `L` unifiziert werden kann. Verwenden Sie dabei abgesehen vom Cut und dem in SWI-Prolog vordefinierten Prädikat `fail/0` keine weiteren Prädikate, und insbesondere nicht `\=/2`.

(c) Führen Sie in `blatt8.pl` einen neuen Operator `<=>` für die Biimplikation $\leftrightarrow$ ein, der den gleichen Typ und die gleiche Präzedenz wie der in `al.pl` definierte Operator `=>` hat.

(d) Implementieren Sie in `blatt8.pl`, analog zu Beispiel 2.54 im Vorlesungsskript, Schritt 1 des Tseitin-Verfahrens. D.h., schreiben Sie ein Prädikat `tseitin/2`, so dass die Anfrage `tseitin(F, L)` für eine aussagenlogische Formel `F` eine Liste `L` aussagenlogischer Formeln ausgibt, die die folgenden Eigenschaften hat:

- Die Konjunktion der Formeln in der Liste `L` ist erfüllbarkeitsäquivalent zu `F`.
- Die Liste `L` enthält für jede Teilformel von `F` (abgesehen von Literalen) genau eine Formel.
- In jeder Formel aus `L` kommen höchstens 3 verschiedene Aussagensymbole vor.

Beispielsweise sollte Prolog auf die Anfrage:

```
tseitin((p => ~q) \/ (~ (p /\ q) /\ r), L).
```

wie folgt antworten:

```
L = [x1, x1<=>x2\/x3, x2<=> (p=> ~q), x3<=>x4/\r, x4<=> ~x5, x5<=>p/\q].
```

Hierbei sind die konkrete Wahl der neuen Aussagensymbole sowie die Reihenfolge der Formeln in der Repräsentation der Menge unwesentlich.

Hinweise:

- Benutzen Sie zur Erzeugung neuer Aussagensymbole das in SWI-Prolog eingebaute Prädikat `gensym/2`. Das Prädikat `gensym/2` instantiiert bei dem Aufruf `gensym(x, A)` die Variable `A` mit einem Atom der Form xn, wobei eine Zahl n so gewählt wird, dass das Atom xn in diesem Lauf von SWI-Prolog noch nicht verwendet wurde.
- Benutzen Sie den in Teilaufgabe (c) definierten Operator `<=>`.
- Nutzen Sie ggf. Cut oder Negation. Führen Sie bei Bedarf Hilfsprädikate ein.