Humboldt-Universität zu Berlin
Lehrstuhl Logik in der Informatik
Dr. Christoph Berkholz

29. Januar 2019

Logik in der Informatik

Wintersemester 2018/2019

Übungsblatt 13

Abgabe: *bis 5. Februar 2019, 11.15 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: (30 Punkte)

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:

(a) Das Unerfüllbarkeitsproblem für $\mathsf{FO}[\sigma]$ ist nicht entscheidbar.

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

(c) Das Folgerungsproblem für $\mathsf{FO}[\sigma]$ ist nicht entscheidbar.

Aufgabe 2: (20 Punkte)

(a) Sei R ein 2-stelliges Relationssymbol, f ein 1-stelliges Funktionssymbol und seien c und d Konstantensymbole.

Im Folgenden ist für jedes $i \in \{1, 2\}$ eine Signatur σ_i und ein $\mathsf{FO}[\sigma_i]$-Satz φ_i gegeben.

(1) Sei $\sigma_1 := \{R, f, c\}$ und sei φ_1 der folgende $\mathsf{FO}[\sigma_1]$-Satz:

$$\forall x \forall y \Big(\big(R(x,y) \;\rightarrow\; y{=}f(x) \big) \;\;\wedge\;\; \big(y{=}f(x) \;\rightarrow\; R(x,y) \big) \Big)$$

(2) Sei $\sigma_2 := \{R, c, d\}$ und sei φ_2 der folgende $\mathsf{FO}[\sigma_2]$-Satz:

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

Geben Sie für jedes $i \in \{1, 2\}$ eine σ_i-Herbrandstruktur $\mathcal{A}_i$ und eine σ_i-Herbrandstruktur $\mathcal{B}_i$ an, so dass gilt:

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

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

(b) Sei $\sigma := \{f, c\}$, wobei f ein 1-stelliges Funktionssymbol ist und c ein Konstantensymbol.

Zeigen Sie, dass Satz 4.46 aus dem Vorlesungsskript im Allgemeinen *nicht* für $\mathsf{FO}[\sigma]$-Sätze in Skolemform gilt, die *nicht* gleichheitsfrei sind.

Geben Sie dazu einen $\mathsf{FO}[\sigma]$-Satz φ in Skolemform an, so dass gilt:

$$\varphi \text{ ist erfüllbar,} \qquad \text{aber} \qquad \varphi \text{ besitzt kein Herbrand-Modell.}$$

Aufgabe 3: (25 Punkte)

Sei $\sigma := \{R, f\}$, wobei R ein 2-stelliges Relationssymbol und f ein 1-stelliges Funktionssymbol ist. Transformieren Sie die $\mathsf{FO}[\sigma]$-Formel

$$\forall x\, \neg\Big(\neg f(x){=}y \quad \vee \quad \forall y\, R(x, y)\Big)$$

in einen zu φ erfüllbarkeitsäquivalenten gleichheitsfreien $\mathsf{FO}[\hat{\sigma}]$-Satz $\hat{\varphi}$ in Skolemform. Gehen Sie dabei vor wie im Beweis von Satz 4.52 im Vorlesungsskript. Geben Sie insbesondere auch die Signatur $\hat{\sigma}$ sowie die Formeln an, die nach jedem der Schritte 1, 2 und 3 des Beweises entstehen.

Aufgabe 4: (25 Punkte)

Lesen Sie Kapitel 8 aus dem Buch „Learn Prolog Now!".

Achtung: *Die Bearbeitung der Aufgabe ist unter Beachtung der bekannten Abgabehinweise über Moodle abzugeben!*

(a) Implementieren Sie ein Prädikat `sat/1`, so dass eine Anfrage

```
?- sat(F).
```

für eine aussagenlogische Formel `F` genau dann erfolgreich ist, wenn `F` erfüllbar ist.

Hinweise: Ihr Prädikat soll zu der Formel `F` zuerst eine *erfüllbarkeitsäquivalente* 3-*KNF* konstruieren, und anschließend deren Erfüllbarkeit mit dem DPLL-Algorithmus testen. Es macht nichts, wenn Ihr Prädikat für eine erfüllbare Formel mehrfach

```
true.
```

ausgibt.

(b) Für Vergleiche von SAT-Solvern werden 3-KNF oft im sogenannten DIMACS-Format angegeben (vgl. `https://www.satcompetition.org/2009/format-benchmarks2009.html`). Implementieren Sie ein Prädikat `sat_dimacs/1`, welche als Argument den Namen einer Datei erhält, so dass beispielsweise die Anfrage

```
?- sat_dimacs('knf.cnf').
```

genau dann erfolgreich ist, falls die in der Datei `knf.cnf` repräsentierte 3-KNF erfüllbar ist. Ist diese nicht erfüllbar, soll das Ergebnis `false.` sein.

Sie können in Ihrer Implementation davon ausgehen, dass die aufgerufene Datei im akutellen Verzeichnis existiert und dem DIMACS-Standard entspricht.

Sie können zur Lösung dieser Aufgabe alle Prolog-Module verwenden, die Sie unter `https://www2.informatik.hu-berlin.de/logik/lehre/WS18-19/Logik/prolog-uebung.shtml` vorfinden. Dies gilt insbesondere für die Module `tseitin.pl` und `dpll.pl`[1]. Dort finden Sie auch Beispieldateien im DIMACS-Format.

[1] Verfügbar ab 29.01.19.