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

28. Januar 2015

Logik in der Informatik

Wintersemester 2014/2015

Übungsblatt 12

Abgabe: *bis 4. Februar 2015, 9.15 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: (24 Punkte)

(a) Welche der beiden folgenden Aussagen ist für jede Signatur σ und jede $\mathsf{FO}[\sigma]$-Formel φ 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 $\sigma = \{E/2\}$. Betrachten Sie die $\mathsf{FO}[\sigma]$-Formel

$$\varphi(x) \;:=\; \neg\, \exists y \forall z \Big(E(y,z) \rightarrow (E(x,z) \wedge \exists x\, E(x,x))\Big).$$

(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.

(c) Beweisen Sie Satz 4.68 aus der Vorlesung, das heißt zeigen Sie:

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

Aufgabe 2: (24 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen:

(a) Sei $\sigma = \emptyset$. Es gibt einen $\mathsf{FO}[\sigma]$-Satz φ_a, so dass für jede σ-Struktur $\mathcal{A}$ gilt:

$$\mathcal{A} \models \varphi_a \quad \Longleftrightarrow \quad |A| \text{ ist eine Primzahl.}$$

(b) Sei $\sigma = \{E/2, F/2\}$ eine relationale Signatur. Es gibt einen $\mathsf{FO}[\sigma]$-Satz φ_b, so dass für jede σ-Struktur $\mathcal{A} = (A, E^{\mathcal{A}}, F^{\mathcal{A}})$, bei der $\mathcal{G} := (A, E^{\mathcal{A}})$ ein endlicher gerichteter Pfad ist, gilt:

$$\mathcal{A} \models \varphi_b \quad \Longleftrightarrow \quad F^{\mathcal{A}} \text{ ist der reflexive und transitive Abschluss von } E^{\mathcal{A}}.$$

Dabei nutzen wir folgende Begriffe:

- Ein Graph $\mathcal{G} = (V^{\mathcal{G}}, E^{\mathcal{G}})$ ist ein endlicher gerichteter Pfad, falls es ein $n \in \mathbb{N}$ mit $n \geq 1$ gibt, so dass $|V^{\mathcal{G}}| = n$, $V^{\mathcal{G}} = \{v_1, \ldots v_n\}$ und $E^{\mathcal{G}} = \{\, (v_i, v_{i+1}) \,:\, 1 \leq i < n \,\}$.
- Seien $E^{\mathcal{A}}, F^{\mathcal{A}} \subseteq A \times A$. $F^{\mathcal{A}}$ heißt transitiver und reflexiver Abschluss von $E^{\mathcal{A}}$, wenn für alle $(a, a') \in A \times A$ gilt:

$$(a, a') \in F^{\mathcal{A}} \quad \Longleftrightarrow \quad a = a' \text{ oder es gibt im gerichteten Graphen } \mathcal{G} = (A, E^{\mathcal{A}}) \text{ einen Weg vom Knoten } a \text{ zum Knoten } a'.$$

Aufgabe 3: (27 Punkte)

Wir betrachten im Folgenden Kalküle über der Menge $M := \mathsf{AL}(\{\neg, \rightarrow\})$.

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

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 $\dfrac{}{\varphi}$.

Außerdem enthält

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

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

  die *Abduktion* genannt wird,

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

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

  die *Syllogismus* genannt wird.

(a) Geben Sie für $\varphi := \neg(V_0 \rightarrow V_0)$ und $\Phi := \emptyset$ eine möglichst kurze Ableitung von φ aus Φ in $\mathfrak{K}_{Abd}$ an.

(b) Zeigen Sie, dass $\mathfrak{K}_{Abd}$ vollständig, aber nicht korrekt ist.

(c) Zeigen Sie, dass $\mathfrak{K}_{Syl}$ korrekt, aber nicht vollständig ist.

(d) Betrachten Sie den Kalkül $\mathfrak{K}$ über M, der alle Ableitungsregeln aus $\mathfrak{K}_{Syl}$ und alle Ableitungsregeln aus $\mathfrak{K}_{Abd}$ enthält. Geben Sie an, ob $\mathfrak{K}$ korrekt bzw. vollständig ist.

Aufgabe 4: (25 Punkte)

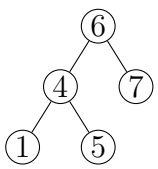
Lesen Sie Kapitel 8 aus dem Buch "Learn Prolog Now!".

In dieser Aufgabe betrachten wir *geordnete und beschriftete Binärbäume*, die folgende Eigenschaften erfüllen:

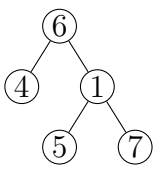
- Jeder Binärbaum enthält mindestens einen Knoten.
- Jeder Knoten in einem Binärbaum ist entweder ein *Blattknoten*, d.h. er besitzt keine Kinderknoten, oder er ist ein *innerer Knoten*, d.h. er besitzt sowohl einen linken als auch einen rechten Teilbaum, die beide nicht leer sind.
- Jeder Knoten in einem Binärbaum ist mit einem Prolog-Term beschriftet.

Betrachten Sie beispielsweise folgende Binärbäume:

Binärbaum 1:



Binärbaum 2:



Wir repräsentieren Binärbäume wie folgt durch Prolog-Terme: Ist `T` ein Prolog-Term, dann entspricht `node(T, nil, nil)` dem Binärbaum, der aus genau einem, mit `T` beschrifteten, Blattknoten besteht. Repräsentieren `L` und `R` Binärbäume, dann ist `node(T, L, R)` der Binärbaum, der aus einem mit `T` beschrifteten inneren Knoten besteht, an den als linker Teilbaum `L` und als rechter Teilbaum `R` angehängt ist. Beispielsweise wird *Binärbaum 1* repräsentiert durch den Term

```
node(6, node(4, node(1, nil, nil), node(5, nil, nil)), node(7, nil, nil))
```

Achtung: *Die Bearbeitung der folgenden Aufgaben ist in einer Datei* `btree.pl` *digital über das GOYA-System abzugeben!*

(a) Eine Liste `X` nennen wir die *pre-order-Traversierung* eines Binärbaum `B`, wenn das folgende Prädikat `preorder/2` die Anfrage `?- preorder(X, B).` erfüllt:

```
preorder(node(T, nil, nil), [T]) :- !.
preorder(node(T, L, R), [T|X]) :-
    preorder(L, LX), preorder(R, RX), append(LX, RX, X).
```

Obwohl jeder Binärbaum eine eindeutige pre-order-Traversierung besitzt, gibt es umgekehrt Listen, die pre-order-Traversierungen von keinem oder auch von mehr als einem Binärbaum sind. Beispielsweise ist die Liste `[6, 4, 1, 5, 7]` sowohl für *Binärbaum 1* als auch für *Binärbaum 2* eine pre-order-Traversierung.

Schreiben Sie eine *Definite Clause Grammar mit einem zusätzlichen Argument*, so dass die Anfrage `?- bt(B, X, []).` für einen Binärbaum `B` und eine Liste `X` genau dann erfüllt ist, wenn `X` eine pre-order-Traversierung ist.

(b) Wir nennen einen Binärbaum einen *binären Suchbaum*, wenn seine Knoten nur mit Ganzzahlen beschriftet sind und für jeden seiner inneren Knoten `node(Z, L, R)` gilt: Jeder Knoten in `L` ist mit einer Zahl $< $ `Z`, und jeder Knoten in `R` mit einer Zahl $>$ `Z` beschriftet. Beispielsweise ist *Binärbaum 1* ein binärer Suchbaum, *Binärbaum 2* jedoch nicht.

Schreiben Sie eine *Definite Clause Grammar mit drei zusätzlichen Argumenten*, so dass die Anfrage `?- bst(B, Min, Max, X, []).` genau dann erfüllt ist, wenn `X` die pre-order-Traversierung für den *binären Suchbaum* `B` ist, und zusätzlich `Min` und `Max` die kleinste, beziehungsweise größte Ganzzahl ist, die als Beschriftung in `B` vorkommt.

Hinweis: *Nutzen Sie die Möglichkeit, Ihrer Definite Clause Grammar mit Hilfe der Notation* $\{\ldots\}$ *zusätzliche Ziele hinzuzufügen. Verwenden Sie gegebenfalls das eingebaute Prädikat* `integer/1`.