Zusammenfassung: Logik für die Informatik

Rico Klimpel

February 7, 2020

Contents

I Aussagenlogik **2**

II Prädikatenlogik **3**

1 Syntax & Semantik **3**
- 1.1 Signatur 3
- 1.2 Struktur 3
- 1.3 Terme 3
- 1.4 Formeln 3
- 1.5 Interpretation von Termen 4
- 1.6 Interpretation von Formlen 4
- 1.7 Freie Variablen 4
- 1.8 Koinzidenzlemma 4

2 Modelierung **4**
- 2.1 Relationen in Strukturen defnieren? 4
- 2.2 Erfüllbarkeit einer Formel 4

3 Äquivalenz **4**
- 3.1 Äquivalenz von Formeln 4
- 3.2 Regeln der Prädikatenlogik 4
- 3.3 Quantorenregeln 4
- 3.4 Umbenennen von gebundenen Variablen 4
- 3.5 Scope von Quantoren 4
- 3.6 Normalformen 5
  - 3.6.1 Boolsche Normalform 5
  - 3.6.2 Plenex Normalform 5
  - 3.6.3 Konjunktive Normalform 5

4 Folgerungsbeziehungen (Entailment) **5**
- 4.1 Folgerungsbeziehung 5
- 4.2 Beziehung zwischen Erfüllbarkeit und Folgerungsbeziehung 5

5 Beweissysteme **5**
- 5.1 Natürliches Beweissystem 5
  - 5.1.1 Beweisregeln 5
  - 5.1.2 Korrektheit & Vollständigkeit 5
- 5.2 Resolutionsbeweise 5
  - 5.2.1 Korrektheit & Vollständigkeit 5
  - 5.2.2 Verbindung zwischen Resolution und Logik-Programmierung 5

6 Kompaktheit **5**

Informationen

Zusammenfassung der Vorlesung Logik für die Informatik an der CAU Kiel aus dem Wintersemester 2019/2020, gehalten von Prof. Dr. Thomas Wilke. Ein Versuch die wichtigsten Aussagen ohne enorm lange Formalitäten drum herum knapp zu Papier zu bringen. Kein Anspruch auf Vollständigkeit. Geschrieben in LaTeX.

Part I
Aussagenlogik

Part II
Prädikatenlogik

1 Syntax & Semantik

1.1 Signatur

`https://lili.informatik.uni-kiel.de/llocs/Signatures`

Eine Signatur $\mathcal{S}$ besteht aus eine Menge S von Symbolen und einer Funktion $\Sigma\colon S \to \mathbb{N} \cup \mathbb{N} \times \{1\}$.
The Elemente von S werden Symbole genannt und wie folgt eingeteilt:

- Ein Symbol f mit $\Sigma(f) = \langle n, 1\rangle$ für $n > 0$ ist eine Funktionssymbol.
  Menge dieser Symbole: $\mathcal{F}_\Sigma$ oder einfach $\mathcal{F}$.
- Ein Symbol R mit $\Sigma(R) = n$ für $n > 0$ ist ein Relationssymbol.
  Menge dieser Symbole: $\mathcal{R}_\Sigma$ oder $\mathcal{R}$.
- Ein Symbol c mit $\Sigma(c) = \langle 0, 1\rangle$ ist ein Symbol für eine Konstante.
  Menge dieser Symbole: $\mathcal{C}_\Sigma$ oder $\mathcal{C}$.
- Symbol b mit $\Sigma(b) = 0$ ist ein Symbol für einen boolschen Wert.
  Menge dieser Symbole: $\mathcal{B}_\Sigma$ oder $\mathcal{B}$.

Im allgemeinen werden Signaturen mit $\mathcal{B} \neq \emptyset$ ignoriert (Signaturen ohne boolsche Werte). Keine Ahnung warum er das sagt.

Beispiele:

$$S = \{\text{zero}, \text{one}, \text{add}, \text{mult}\}$$
$$\Sigma = \{\text{zero} \mapsto \langle 0, 1\rangle, \text{one} \mapsto \langle 0, 1\rangle, \text{add} \mapsto \langle 2, 1\rangle, \text{mult} \mapsto \langle 2, 1\rangle\}$$

Vereinfacht aufgeschrieben sieht das ganze so aus:

$$\mathcal{S} = \{\text{zero}, \text{one}, \text{add}//2, \text{mult}//2\}$$

1.2 Struktur

`https://lili.informatik.uni-kiel.de/llocs/Structures`

Sei $\mathcal{S}$ eine Signatur. Eine $\mathcal{S}$-Struktur $\mathcal{A}$ besteht aus:

- Univserum A mit $A \neq \emptyset$
- Für jedes Symbol eine Konstanten $c \in \mathcal{S}$ eine Interpretation $c^{\mathcal{A}} \in A$ von c.
- Für jedes Funktionssymbol $f//n \in \mathcal{S}$ eine Interpretation $f^{\mathcal{A}}\colon A^n \to A$
- Für jedes Relationssymbol $R/n \in \mathcal{S}$ eine Interpretation $R^{\mathcal{A}} \subseteq A^n$

Hier ein Beispiel das ungefähr zu der Signatur oben passt:

$$\begin{aligned}
A &= \{0, 1, 2, 3\} \\
\text{zero}^{\mathcal{A}} &= 3 \\
\text{one}^{\mathcal{A}} &= 2 \\
\text{add}^{\mathcal{A}}(a, b) &= 0 && \text{for } a, b \in A \\
\text{mult}^{\mathcal{A}}(a, b) &= a + b \text{ rest } 4 && \text{for } a, b \in A \\
\text{Lt}^{\mathcal{N}} &= \{\langle a, a\rangle : a \in A\}
\end{aligned}$$

1.3 Terme

`https://lili.informatik.uni-kiel.de/llocs/Syntax_of_first-order_logic#Formal_definition_of_terms`

Induktive Defintion für alle Terme über eine Signatur $\mathcal{S}$, die auch $\mathcal{S}$-terms genannt wird: Basiselemente:

- Ein Baum mit nur einem Element das eine Variable der Prädikatenlogik enthält ist ein $\mathcal{S}$-term.
- Ein Baum mit nur einem Element das eine Konstante $c \in \mathcal{S}$ enthält ist ein $\mathcal{S}$-term.

Diese werden die atomaren $\mathcal{S}$-terme genannt. Induktionsregeln:

- Wenn $f//n \in \mathcal{S}$ eine Funktion und $t_0, \dots, t_{n-1}$ $\mathcal{S}$-terms sind, dann ist der Baum mit der Wurzel f und den n Teilbäumen t_0, ..., t_{n-1} ein $\mathcal{S}$-term.

1.4 Formeln

`https://lili.informatik.uni-kiel.de/llocs/Syntax_of_first-order_logic#Formal_definition_of_formulas`

Induktive Defintion für alle Formeln über eine Signatur $\mathcal{S}$, die auch $\mathcal{S}$-formulas genannt wird: Basiselemente:

- Der einelementige Baum in dem das einzige Element eines der konstanten Symbole $\top$ oder $\perp$ ist, ist eine (prädikatenlogische) Formel.
- Wenn t_0, t_1 Terme sind, dann ist der Baum mit der Wurzel $\doteq$ und den Teilbäumen t_0 und t_1 eine Formel.
- Wenn $R/n \in \mathcal{S}$ eine Relation ist und $t_0, \dots, t_{n-1}$ Terme sind dann ist der Baum mit der Wurzel R und den n Teilbäumenm $t_0, \dots, t_{n-1}$ eine Formel.

Diese werden die atomaren Formeln genannt. Induktionsregeln:

- Wenn C ein n-stelliger Junktor ist und $\varphi_0, \dots, \varphi_{n-1}$ Formeln sind, dann ist der Baum mit der Wurzel C und den n Teilbäumen $\varphi_0, \dots, \varphi_{n-1}$ eine Formel.
- Wenn x_i eine Variable ist und φ eine Formel, dann ist der Baum mit der Wurzel $\exists x_i$ oder der Wurzel $\forall x_i$ und dem Teilbaum φ eine Formel.

1.5 Interpretation von Termen

`https://lili.informatik.uni-kiel.de/llocs/Semantics_of_first-order_logic#Interpretation_of_terms`

Sei $\mathcal{S}$ eine Signatur und $\mathcal{A}$ eine $\mathcal{S}$-Struktur. Für eine Belegung (A-Belegung) β, ist der Wert von jedem $\mathcal{S}$-term t in $\mathcal{A}$ unter β: $[\![t]\!]^{\mathcal{A}}_{\beta}$ defniert durch folgender Induktion. Basiselemente:

- Für alle $i \in \mathbb{N}$ gilt: $[\![x_i]\!]^{\mathcal{A}}_{\beta} = \beta(x_i)$.
  Variablen bekommen den ihnen unter β zugewiesenen Wert bei der alleinigen Auswertung.
- Für jedes $c \in C$ gilt: $[\![c]\!]^{\mathcal{A}}_{\beta} = c^{\mathcal{A}}$
  Konstante Symbole werden wie in der Struktur beschriegben ausgwertet wenn sie alleine stehen.

Induktionsregel:

- Für alle $f//n \in \mathcal{F}$ und die $\mathcal{S}$-terms $t_0, \ldots, t_{n-1}$ gilt: $[\![f(t_0, \ldots, t_{n-1})]\!]^{\mathcal{A}}_{\beta} = f^{\mathcal{A}}([\![t_0]\!]^{\mathcal{A}}_{\beta}, \ldots, [\![t_{n-1}]\!]^{\mathcal{A}}_{\beta})$

1.6 Interpretation von Formlen

`https://lili.informatik.uni-kiel.de/llocs/Semantics_of_first-order_logic#Interpretation_of_formulas`

So dieses mal einfach direkt die Induktive Definition: Basiselemente:

- Für die boolschen konstannten Symbole gilt:

$$[\![\bot]\!]^{\mathcal{A}}_{\beta} = 0 \tag{1}$$
$$[\![\top]\!]^{\mathcal{A}}_{\beta} = 1 \tag{2}$$

- Für alle Terme t_0, t_1 gilt:

$$[\![t_0 \doteq t_1]\!]^{\mathcal{A}}_{\beta} = \begin{cases} 1 & \text{if } [\![t_0]\!]^{\mathcal{A}}_{\beta} = [\![t_1]\!]^{\mathcal{A}}_{\beta} \\ 0 & \text{sonst} \end{cases} \tag{3}$$

- Für alle Relationen $R/n \in \mathcal{R}$ und Terme $t_0, \ldots, t_{n-1}$ gilt:

$$[\![R(t_0, \ldots, t_n)]\!] = \begin{cases} 1 & \text{if } \langle [\![t_0]\!]^{\mathcal{A}}_{\beta}, \ldots, [\![t_{n-1}]\!]^{\mathcal{A}}_{\beta} \rangle \in R^{\mathcal{A}} \\ 0 & \text{sonst} \end{cases} \tag{4}$$

Induktionsregeln:

- Für jeden n-stelligen Junktor C und die Formlen $\varpi_0, ..., \varphi_{n-1}$ gilt:

$$[\![C(\varphi_0, \ldots, \varphi_{n-1})]\!]^{\mathcal{A}}_{\beta} = f_C([\![\varphi_0]\!]^{\mathcal{A}}_{\beta}, \ldots, [\![\varphi_{n-1}]\!]^{\mathcal{A}}_{\beta})$$

- Für jede Formel φ und $i \in \mathbb{N}$ gilt:

$$[\![\exists x_i \varphi]\!]^{\mathcal{A}}_{\beta} = \begin{cases} 1 & \text{wenn ein } a \in A \text{ existiert,} \\ & \text{für das gilt } [\![\varphi]\!]^{\mathcal{A}}_{\beta[\frac{x_i}{a}]} = 1 \\ 0 & \text{sonst} \end{cases}$$

$$[\![\forall x_i \varphi]\!]^{\mathcal{A}}_{\beta} = \begin{cases} 1 & \text{wenn für alle } a \in A \text{ gilt:} \\ & [\![\varphi]\!]^{\mathcal{A}}_{\beta[\frac{x_i}{a}]} = 1 \\ 0 & \text{sonst} \end{cases}$$

$[\![\varphi]\!]^{\mathcal{A}}_{\beta} = 1$ wird auch als $\mathcal{A}, \beta \models \varphi$

1.7 Freie Variablen

`https://lili.informatik.uni-kiel.de/llocs/Free_variables`

1.8 Koinzidenzlemma

`https://lili.informatik.uni-kiel.de/llocs/Coincidence_lemma_(first-order_logic)`

2 Modelierung

2.1 Relationen in Strukturen defnieren?

`https://lili.informatik.uni-kiel.de/llocs/Definable_relations`

2.2 Erfüllbarkeit einer Formel

`---`

3 Äquivalenz

3.1 Äquivalenz von Formeln

`https://lili.informatik.uni-kiel.de/llocs/Formula_equivalence_(first-order_logic)`

3.2 Regeln der Prädikatenlogik

`---`

3.3 Quantorenregeln

`---`

3.4 Umbenennen von gebundenen Variablen

`---`

3.5 Scope von Quantoren

`https://lili.informatik.uni-kiel.de/llocs/Scope_of_quantifiers_in_first-order_logic`

3.6 Normalformen

3.6.1 Boolsche Normalform

https://lili.informatik.uni-kiel.de/llocs/Boolean_normal_form_(first-order_logic)

3.6.2 Plenex Normalform

https://lili.informatik.uni-kiel.de/llocs/Prenex_normal_form

3.6.3 Konjunktive Normalform

https://lili.informatik.uni-kiel.de/llocs/Conjunctive_normal_form_(first-order_logic)

4 Folgerungsbeziehungen (Entailment)

4.1 Folgerungsbeziehung

4.2 Beziehung zwischen Erfüllbarkeit und Folgerungsbeziehung

---

5 Beweissysteme

5.1 Natürliches Beweissystem

5.1.1 Beweisregeln

5.1.2 Korrektheit & Vollständigkeit

5.2 Resolutionsbeweise

5.2.1 Korrektheit & Vollständigkeit

5.2.2 Verbindung zwischen Resolution und Logik-Programmierung

---

6 Kompaktheit
