Theoretische Informatik II

Dr. Eva Richter / Holger Arnold
Universität Potsdam, Theoretische Informatik, Sommersemester 2008

Übungsblatt 8 (Version 3) — Abgabetermin: 23.6.2008, 12.00 Uhr

---

Formale Systeme

Dieser Abschnitt enthält eine (sehr) knappe Einführung zu formalen Systemen.

Definition 1. Ein *formales System* $F = (L, R, A)$ besteht aus

1. einer entscheidbaren *formalen Sprache* L, die beschreibt, welche Formeln in F gebildet werden können; diese bezeichnet man als *wohlgeformte Formeln*;
2. einer Menge R von mindestens zweistelligen *Ableitungsrelationen* auf Formeln, die beschreibt, welche Formeln durch F ineinander überführt werden können;
3. einer Menge $A \subseteq L$ von *Axiomen*, die auch leer sein kann.[1]

Ein *Ableitung* einer Formel ϕ aus einer Formelmenge Γ in F ist eine Folge $\phi_1, \phi_2, \dots, \phi_k$, so dass $\phi = \phi_k$ und für alle $i \in \{1, \dots, k\}$ gilt: $\phi_i \in A$ oder $\phi_i \in \Gamma$ oder $(\Delta, \phi_i) \in r$ für eine Teilmenge $\Delta \subseteq \{\phi_1, \dots, \phi_{i-1}\}$ und eine Relation $r \in R$. Existiert eine Ableitung von ϕ aus Γ in F, dann heißt ϕ *aus* Γ *ableitbar* und man schreibt $\Gamma \vdash_F \phi$. Ist ϕ aus der leeren Menge ableitbar, dann heißt ϕ *ableitbar in* F oder *Theorem in* F und man schreibt $\vdash_F \phi$. Eine Ableitung von ϕ aus $\emptyset$ heißt *Beweis für* ϕ. Ein formales System F heißt *konsistent*, wenn mindestens eine Formel nicht in F ableitbar ist.

Beispiel 2 (Turingmaschinen)**.** *Sei* $M = (Q, \Sigma, \Gamma, \delta, q_0, q_A, q_R)$ *eine Turingmaschine und sei* $\mathrm{TM}_M = (L, R, A)$ *das folgende formale System:*

1. $L = \{(v, q, w) \mid q \in Q, v, w \in \Gamma^*\}$ *ist die Menge aller wohlgeformten Konfigurationen von* M*. Beachten Sie, dass die Konfigurationen in* L *nicht alle tatsächlich erreichbar sein müssen.*
2. $R = \{\{(c_1, c_2) \mid c_1, c_1 \in L \text{ und } c_1 \vdash_M c_2\}\}$ *ist die Übergangsrelation für Konfigurationen von* M*. Auch für die Definition von* R *ist nicht relevant, ob die in* R *enthaltenen Konfigurationen tatsächlich erreicht werden können.*
3. $A = \{(\epsilon, q_0, w) \mid w \in \Sigma^*\}$ *ist die Menge der Startkonfigurationen von* M*.*

Die Menge aller in TM_M *ableitbaren Formeln ist gerade die Menge aller Konfigurationen, die von* M *auf einer Eingabe erreicht werden können. Ein „Beweis" einer Konfiguration* c *in* TM_M *ist eine gültige Folge von Konfigurationen, die mit einer Startkonfiguration beginnt und mit* c *endet.*

Die Ableitungsrelationen und Axiome eines formalen Systems werden häufig in Form von *Ableitungsregeln* dargestellt. Diese besitzen die Form

$$„(\mathsf{R})\ \frac{A_1 \quad \cdots \quad A_n}{B(A_1, \dots, A_n)} \quad \text{falls } P(A_1, \dots, A_n) \text{ gilt.“}$$

[1] Die Axiomenmenge kann auch als einstellige Ableitungsrelation betrachtet werden.

Die Elemente $A_1, \ldots, A_n$ und B der Regel R sind technisch gesehen *keine* Formeln, sondern Schemata für noch einzusetzende Formeln. Die Regel R ist auf Formeln $\phi_1, \ldots, \phi_n$ *anwendbar*, wenn diese dem von $A_1, \ldots, A_n$ vorgegebenen Schema entsprechen und die Bedingung $P(\phi_1, \ldots, \phi_n)$ erfüllt ist. Das Resultat der Regelanwendung ist die Formel $B(\phi_1, \ldots, \phi_n)$. Die Regel R definiert folgende $(n+1)$-stellige Ableitungsrelation r:

$$r = \{(\phi_1, \ldots, \phi_n, \phi) \mid \text{die Regel } \mathsf{R} \text{ ist auf } \phi_1, \ldots, \phi_n \text{ anwendbar und das Resultat ist } \phi\}.$$

Beispiel 3. *Eine bekannte Ableitungsregel in der Logik ist die Abtrennungsregel („modus ponens“):*

$$(\mathsf{MP}) \quad \frac{A \qquad A \to B}{B}$$

Diese Regel entspricht der Schlussfolgerung, dass, wenn die Aussage ϕ wahr ist und aus ϕ die Aussage ψ folgt, dann auch ψ wahr ist. Die Regel (MP) *ist anwendbar auf die Formeln $(a \wedge b)$ und $(a \wedge b) \to (c \vee \neg d)$; das Resultat der Regelanwendung ist $(c \vee \neg d)$. Dagegen ist* (MP) *nicht anwendbar auf die Formeln a und $(b \wedge c)$, weil diese nicht dem Regelschema entsprechen (der Hauptoperator der zweiten Formel muss „$\to$“ sein).*

Beispiel 4 (λ-Gleichheit)**.** *Das folgende formale System $\mathcal{E}_\lambda = (L, R, A)$ haben Sie bereits in Übung 3 kennengelernt. Es formalisiert die λ-Gleichheit $\approx$ im λ-Kalkül und ist wie folgt definiert:*

1. *$L = \{t_1 \approx t_2 \mid t_1$ und t_2 sind wohlgeformte λ-Terme$\}$.*
2. *R ist die durch folgende Ableitungsregeln definierte Menge von Ableitungsrelationen:*

$$(\mathsf{Sym}) \ \frac{s \approx t}{t \approx s} \qquad (\mathsf{Trans}) \ \frac{s \approx t \qquad t \approx u}{s \approx u}$$

$$(\mathsf{Subst}_1) \ \frac{s \approx t}{u\ s \approx u\ t} \qquad (\mathsf{Subst}_2) \ \frac{s \approx t}{\lambda x \cdot s \approx \lambda x \cdot t}$$

3. *A ist die durch folgende Regeln definierte Menge von Axiomen:*

$$(\mathsf{Konv}) \ \frac{}{s \approx t} \ \textit{falls } s \equiv t \qquad (\mathsf{Red}) \ \frac{}{s \approx t} \ \textit{falls } s \longrightarrow t$$

$$(\mathsf{Refl}) \ \frac{}{t \approx t}$$

Die Menge aller in $\mathcal{E}_\lambda$ ableitbaren Formeln ist gerade die Menge aller λ-Gleichheiten zwischen λ-Termen. Das System $\mathcal{E}_\lambda$ ist konsistent, denn zum Beispiel ist $\mathsf{true} \approx \mathsf{false}$ *nicht ableitbar.*

Soll mit einem formalen System logisches Schließen modelliert werden, muss die Sprache des Systems Formeln für die Begriffe „wahr“ und „falsch“ enthalten; diese werden üblicherweise durch die Formeln $\top$ und $\bot$ dargestellt. Ein solches System heißt dann konsistent, wenn die Formel $\bot$ nicht ableitbar ist. In der klassischen Logik ist diese Bedingung gleichbedeutend damit, dass für keine Formel ϕ sowohl ϕ als auch die Negation $\neg\phi$ abgeleitet werden kann. Ein formales logisches System heißt *korrekt*, wenn nur Formeln abgeleitet werden können, die gültig, also „wahr“ im Sinne der entsprechenden Logik sind. Wenn alle gültigen Formeln abgeleitet werden können, heißt das System *vollständig*.

Beispiel 5 (Hilbert-Kalkül)**.** *Sei HK $= (L, R, A)$ das folgende formale System:*

1. *L ist die Menge aller wohlgeformten aussagenlogischen Formeln über der Variablenmenge $\{x_1, x_2, \ldots\}$, den Operatoren $\neg$ und $\to$ und den Formeln $\top$ und $\bot$.*

2. *R enthält nur die von der Ableitungsregel* (MP) *aus Beispiel 3 definierte Ableitungsrelation.*

3. *A ist die durch folgende Regeln definierte Menge von Axiomen:*

$$(\mathsf{A}_1)\ \frac{}{A \to (B \to A)} \qquad (\mathsf{A}_2)\ \frac{}{(\neg A \to \neg B) \to (B \to A)}$$

$$(\mathsf{A}_3)\ \frac{}{(A \to (B \to C)) \to ((A \to B) \to (A \to C))}$$

Die Menge aller in HK *ableitbaren Formeln ist gerade die Menge aller Formeln über* $\neg$ *und* $\to$, *die in der klassischen Aussagenlogik gültig, also wahr sind. Dieses System ist also korrekt und vollständig für diese Logik. Außerdem ist* HK *konsistent, denn für keine Formel* ϕ *kann sowohl* ϕ *als auch* $\neg\phi$ *abgeleitet werden; insbesondere ist* $\perp$ *nicht ableitbar.*

Der Rekursionssatz

Satz 6 (Rekursionssatz)**.** *Sei* T *eine Turingmaschine und sei* $t : \Sigma^* \times \Sigma^* \to \Sigma^*$ *die von ihr berechnete Funktion. Dann lässt sich aus* T *eine Turingmaschine* R *konstruieren, die eine Funktion* $r : \Sigma^* \to \Sigma^*$ *berechnet, so dass für alle* $x \in \Sigma^*$ *gilt:* $r(x) = t(\langle R \rangle, x)$.

Satz 7 (Fixpunktsatz)**.** *Sei* $t : \Sigma^* \to \Sigma^*$ *eine total berechenbare Funktion. Dann gibt es eine Turingmaschine* F, *so dass* $t(\langle F \rangle)$ *die Beschreibung einer zu* F *äquivalenten Turingmaschine ist.*

Aufgabe 8.1

Eine Turingmaschine M heiße minimal, wenn es keine Turingmaschine mit einer kürzeren Beschreibung gibt, welche die gleiche Sprache akzeptiert wie M. Beweisen Sie, dass die Sprache $MIN_{\mathsf{TM}} = \{\langle M \rangle \mid M \text{ ist eine minimale Turingmaschine}\}$ nicht Turing-akzeptierbar ist.

Hinweis: Nehmen Sie an, es gäbe einen Aufzähler für MIN_{TM}. Konstruieren Sie mit Hilfe des Rekursionssatzes eine Turingmaschine, die sich äquivalent zu einer minimalen Turingmaschine verhält, aber eine kürzere Beschreibung besitzt als diese. Erinnern Sie sich daran, dass nach dem Rekursionssatz eine Turingmaschine M mit ihrer eigenen Beschreibung $\langle M \rangle$ rechnen kann.

Aufgabe 8.2

Für eine Turingmaschine M sei $\phi_{\langle M \rangle}$ die von M berechnete Funktion. Beweisen Sie folgenden Satz:

Satz 8 (S_{mn}-Satz)**.** *Sei $f : \Sigma^* \times \Sigma^* \to \Sigma^*$ eine partiell berechenbare Funktion. Dann gibt es eine total berechenbare Funktion $g : \Sigma^* \to \Sigma^*$, so dass für alle $x, y \in \Sigma^*$ gilt: $f(x, y) = \phi_{g(x)}(y)$.*

Aufgabe 8.3

Sei F ein formales logisches System, das folgende Bedingungen erfüllt:

1. F ist korrekt; es lassen sich also nur „wahre“ Aussagen ableiten.
2. Die Menge $\Pi_F = \{\pi \mid \text{es gibt eine Formel } \phi\text{, so dass } \pi \text{ ein Beweis für } \phi \text{ in } F \text{ ist}\}$ der Beweise in F ist entscheidbar.
3. F ist ausdrucksstark genug, um in der Sprache von F Aussagen über Turingmaschinen zu formulieren. Insbesondere lassen sich Aussagen der Form „Die Turingmaschine M hält auf dem Wort w an“ als Formeln ausdrücken.

Beweisen Sie:

1. Für eine gegebene Formel ϕ und eine Ableitung π ist entscheidbar, ob π ein Beweis für ϕ in F ist.
2. Es gibt eine Turingmaschine G, die eine Funktion $g : \Sigma^* \times \Sigma^* \to \Sigma^*$ berechnet, so dass gilt:

$$g(x, w) = \begin{cases} 1 & \text{wenn } x = \langle M \rangle \text{ für eine Turingmaschine } M \text{ ist und es in} \\ & F \text{ einen Beweis dafür gibt, dass } M \text{ auf } w \text{ nicht anhält,} \\ \text{undefiniert} & \text{sonst.} \end{cases}$$

3. Es gibt eine Turingmaschine K, so dass es für alle Wörter w in F weder einen Beweis dafür gibt, dass K auf w anhält, noch einen Beweis dafür, dass K auf w nicht anhält.

Hinweis: Verwenden Sie Satz 8 aus der vorhergehenden Aufgabe und den Fixpunktsatz.

Hausaufgabe 8.4

Für eine partiell berechenbare Funktion $f : \mathbb{N} \to \mathbb{N}$ sei $\bar{f}$ definiert als

$$\bar{f}(x) = \begin{cases} f(x) & \text{falls } f \text{ an der Stelle } x \text{ definiert ist,} \\ 0 & \text{sonst.} \end{cases}$$

Die Funktion $\bar{f}$ ist offensichtlich auf $\mathbb{N}$ überall definiert. Beweisen Sie, dass die Funktion $\bar{f}$ nicht für jede partiell berechenbare Funktion f berechenbar sein kann.

Hausaufgabe 8.5

Sei t eine Transformation, die in einer Turingmaschinen-Beschreibung die Zustände q_A und q_R vertauscht. Geben Sie einen Fixpunkt der Transformation t an, also eine Turingmaschine M, so dass $t(\langle M \rangle)$ die Beschreibung einer Turingmaschine ist, welche die gleiche Sprache akzeptiert wie M.

Hausaufgabe 8.6

Geben Sie eine Turingmaschine an, die eine Folge $\langle M_1 \rangle, \langle M_2 \rangle, \ldots$ von Beschreibungen von Turingmaschinen aufzählt, so dass jede Turingmaschine in der Aufzählung vorkommt und keine zwei aufeinanderfolgenden Turingmaschinen M_i und M_{i+1} die gleiche Funktion berechnen, oder beweisen Sie, dass es eine solche Turingmaschine nicht geben kann.

Hinweis: Verwenden Sie den Fixpunktsatz.

Hausaufgaben Für jede Hausaufgabe können Sie maximal 3 Punkte bekommen. Die Punkte werden nach folgenden Regeln vergeben:

3 Punkte	= die Aufgabe wurde vollständig und fehlerfrei gelöst
2 Punkte	= die Aufgabe wurde vollständig und im Wesentlichen richtig gelöst, die Lösung enthielt aber einige technische Fehler oder Ungenauigkeiten
1 Punkt	= die Lösung war unvollständig oder enthielt größere Fehler
0 Punkte	= die Aufgabe wurde nicht gelöst

Sprechzeiten Haben Sie Fragen, Anregungen oder Probleme? Lassen Sie es uns wissen!

- Sprechen Sie in den Übungen Ihre Tutorin bzw. Ihren Tutor an.
- Holger Arnold, Raum 1.21, holger@cs.uni-potsdam.de
  Sprechzeiten: mittwochs 14.00–15.00 Uhr und nach Vereinbarung
- Dr. Eva Richter, Raum 1.25, erichter@cs.uni-potsdam.de
  Sprechzeiten: dienstags 13.30–15.00 Uhr und nach Vereinbarung