



I. Aussagenlogik

Logik

Logik ist allgegenwärtig in der Informatik: Jede mathematische Beschreibung eines Algorithmus, jede Spezifikation von Programmen oder Komponenten, jede Datenbank-Anfrage etc. benutzt explizit die Sprache der Logik als Beschreibungsmittel. Es gibt aber noch viele weitergehendere Anwendungen: Insbesondere in sicherheitsrelevanten Situationen kann es angemessen sein, die Korrektheit eines Programms oder eines Algorithmus *formal* zu beweisen; insbesondere bei der Untersuchung kryptographischer Protokolle wird Logik zur Zeit intensiv eingesetzt. Bei verteilten Systemen werden Prozess-Kalküle — selbst eine Art Logik — als Beschreibung verwendet, die dann z.B. mit temporaler Logik studiert werden. Es gibt auch so genannte Logik-Programmiersprachen wie z.B. PROLOG, die automatische Beweissuche in einem Logik-Kalkül direkt in ein Programmier-Paradigma übersetzen. Desweiteren ist *intuitionistische Logik*, eine konstruktive Variante der klassischen Logik, sehr eng mit *funktionalem Programmieren* verbunden, und die logische Untersuchung der Semantik von Programmiersprachen gibt oft wichtige Impulse für die Entwicklung neuer Programmiersprachen bzw. die Erweiterung von bereits bestehenden.

In dieser Vorlesung können wir natürlich nur die Grundlagen für diese weiterführenden Anwendungen legen. Dazu werden wir vor allem *klassische Aussagenlogik* und *Prädikatenlogik* formal untersuchen und, so weit die Zeit es zulässt, auf einige der oben genannten Verbindungen am Rand eingehen.

Das vorgehen der **formalen Logik** ist immer ähnlich; man stellt insbesondere folgende Fragen: Was ist der formale Gegenstand der Untersuchung, also was ist die **Syntax** der Sprache? Was ist die **Bedeutung** von syntaktischen korrekten Ausdrücken? Das ist die Frage nach der **Semantik**, und vor allem möchte man die Semantik einer Aussage **kompositional** aus ihren konstituierenden Teilen erklären. Für die meisten Logiken hat man auch einen formalen, d.h. syntaktischen, Begriff von **Beweis**. Die formale Logik untersucht dann die Verbindung zwischen Syntax, Semantik und Beweis.

Wir wollen einige dieser Begriffe an zwei Beispielen verdeutlichen:

Beispiel: Gleichungslogik

In der Vorlesung ALLGEMEINE ALGEBRA FÜR INFORMATIK haben wir bereits *Gleichungslogik* betrachtet. Dazu haben wir zunächst definiert, was wir überhaupt formal unter einer Gleichung verstehen: ein Element aus $T_{\Sigma}(V) \times T_{\Sigma}(V)$, das wir üblicherweise $s = t$ geschrieben haben, wobei $s, t \in T_{\Sigma}(V)$ Terme sind.

Als nächstes haben wir erklärt, was es heißt, dass eine Gleichung $s = t$ in einer Algebra A vom Typ Σ gilt: Für jede Variablenbelegung $\rho: V \rightarrow A$ gilt $\llbracket s \rrbracket \rho = \llbracket t \rrbracket \rho$. Dann haben wir einen semantischen Folgerungsbegriff $E \models s = t$ erklärt, der besagt, dass $s = t$ in allen Algebren gilt, die alle Gleichungen aus $E \subseteq T_{\Sigma}(V) \times T_{\Sigma}(V)$ erfüllen.

Schließlich haben wir einen syntaktischen Beweisbegriff angegeben, den man knapp mit “Schließen durch Gleichungsketten” umreißen kann, und dann gezeigt (siehe Korollar A.14.⁷¹), dass sich mit diesem “Gleichungskalkül” genau die semantischen Konsequenzen von E beweisen lassen, also dass diese “Gleichungslogik” **korrekt** und **vollständig** ist.

Beispiel: Mathematische Aussagen

Betrachten wir nun ein paar typische mathematische Aussagen:

¹Alle Referenzen der Form “A...” beziehen sich auf das Skript ALLGEMEINE ALGEBRA FÜR INFORMATIK aus dem vergangenen Semester.

- $A \subseteq B \wedge B \subseteq C \rightarrow A \subseteq C$
- $\exists k \in \mathbb{N}. 2 \times k = n$
- $\forall x. \forall y. x \cdot y = 0 \rightarrow (x = 0 \vee y = 0)$
- $\forall \epsilon > 0. \exists \delta > 0. \forall x'. |x' - x| < \delta \rightarrow |f(x') - f(x)| < \epsilon$

In diesen Aussagen kommen sehr unterschiedliche Komponenten vor: Variablen ($A, B, C, k, n, x, y, \epsilon, \delta, x', f$), Operationssymbole ($\times, | \cdot |$), Relationssymbole ($\subseteq, =, <$), Quantoren (\forall, \exists) und **logische Junktoren** ($\wedge, \rightarrow, \vee$). Das interessante an diesen Aussage, vom logischen Standpunkt aus, ist es zu erklären, wie sich die Bedeutung der Gesamtaussage aus ihren Komponenten ergibt. Wir beginnen dazu mit den einfachsten Komponenten, den logischen Junktoren, also den rein logischen “Operationen”.

1 Aussagenlogik

Aussagenlogik untersucht was man über eine Aussage rein aufgrund ihrer logischen Struktur, d.h. unabhängig von ihrem konkreten Gehalt, sagen kann. Zunächst müssen wir aber erklären, was wir überhaupt unter einer **Aussage** verstehen wollen. Die Grundidee ist, dass es sich dabei um einen Satz handelt, von dem es Sinn macht zu sagen, er sei wahr oder falsch. Beispiele dafür sind “Heute ist Dienstag.”, “121 ist prim.”, “Wenn der Mond aus Käse ist, dann ist er blau”. *Keine Aussagen* dagegen sind “Pause!” oder “Was kommt in der Klausur dran?”.

Bei umgangssprachlichen Ausdrücken fällt es manchmal schwer zu entscheiden, ob es sich um einen Ausdruck handelt oder nicht, aber genau davon abstrahiert die *formale* Logik, wie wir gleich sehen werden. Aussagen werden aufgebaut aus **atomaren** Aussagen, wie z.B. “121 ist prim.”, für die wir später einfach Aussagenvariablen schreiben werden, und den logischen **Junktoren** \neg (**Negation**/nicht), \wedge (**Konjunktion**/und), \vee (**Disjunktion**/oder), \rightarrow (**Implikation**/impliziert), \leftrightarrow (**Äquivalenz**/genau dann wenn), \top (**wahr**) und \perp (**Falsum**/Widerspruch).

2 Syntax

In der (klassischen) Aussagenlogik kann man viele der Junktoren durch andere ausdrücken, z.B. kann man sich auf \perp und \rightarrow beschränken. Das entspricht aber zum einen nicht der mathematischen Praxis und passt auch nicht zu dem Beweiskalkül, den wir ab nächster Woche verwenden werden. Wir benutzen deshalb die Junktoren $\perp, \wedge, \vee, \rightarrow$ und wir werden später sehen, wie wir damit \neg und \leftrightarrow ausdrücken können.

Definition 1.1. Wir beginnen mit einer (abzählbar) unendlichen Menge von atomaren Aussagen

$$\mathbf{ATOM} := \{p_0, p_1, p_2, \dots\}.$$

Darauf aufbauend definieren wir *induktiv* die Menge **PROP** aller Aussagen (oder **Propositionen**):

1. Jede atomare Aussage $p_n \in \mathbf{ATOM}$ ist eine Proposition $p_n \in \mathbf{PROP}$.
2. Falsum $\perp \in \mathbf{PROP}$ ist eine Proposition.
3. Sind $A, B \in \mathbf{PROP}$ Proposition, dann auch $(A \wedge B), (A \vee B), (A \rightarrow B) \in \mathbf{PROP}$.

Bemerkung. Um die Lesbarkeit zu erhöhen, lassen wir gelegentlich Klammern — insbesondere die äußersten — weg. Rein formal sind Propositionen aber immer *vollständig geklammerte* Ausdrücke.

2.1 Termalgebra

In der Sprache der Allgemeinen Algebra können wir PROP auch wie folgt verstehen. Wir betrachten die Signatur Σ mit den Operationssymbolen $\Omega = \{\perp, \wedge, \vee, \rightarrow\}$ und den Stelligkeiten $\alpha(\perp) = 0$ und $\alpha(\wedge) = \alpha(\vee) = \alpha(\rightarrow) = 2$. Dann ist PROP bis auf die Verwendung von Infix-Notation einfach die Termalgebra $T_\Sigma(\mathbf{ATOM})$.

2.2 Abgeleitete Operationen

Mit den gegebenen Junktoren erklären wir **Äquivalenz** und **Negation** wie folgt:

Definition 1.2. Seien $A, B \in \text{PROP}$ Propositionen. Wir erklären die syntaktischen Ausdrücke $(A \leftrightarrow B)$ sowie $(\neg A)$ wie folgt als abkürzende Schreibweisen:

$$\begin{aligned}(A \leftrightarrow B) &\equiv ((A \rightarrow B) \wedge (B \rightarrow A)) \\ (\neg A) &\equiv (A \rightarrow \perp)\end{aligned}$$

3 Semantik

Die Grundidee für die Bedeutung einer Proposition $A \in \text{PROP}$ ist, dass man ihr einen **Wahrheitswert** zuordnet. Dies ist, je nach der Bedeutung der atomaren Aussagen p_i , die in A vorkommen, entweder “wahr” oder “falsch”. Wir können deshalb die Semantik $\llbracket A \rrbracket \rho$ immer nur relativ zu einer *Belegung* der atomaren Aussagen definieren.

3.1 Boolesche Algebren

Wie interpretieren daher Propositionen in der booleschen Algebra $\{\mathbf{w}, \mathbf{f}\}$ aus Definition A.1.1, wobei wir die Junktoren \wedge und \vee durch die entsprechenden booleschen Operationen, \perp durch \mathbf{f} und \rightarrow durch die Operation

\rightarrow	\mathbf{w}	\mathbf{f}
\mathbf{w}	\mathbf{w}	\mathbf{f}
\mathbf{f}	\mathbf{w}	\mathbf{w}

interpretieren. Formal können wir das wie folgt ausdrücken:

Definition 1.3. Sei $\rho : \text{ATOM} \rightarrow \{\mathbf{w}, \mathbf{f}\}$ eine *Belegung* der atomaren Aussagen. Dann definieren wir *induktiv* eine Funktion $\llbracket \cdot \rrbracket \rho : \text{PROP} \rightarrow \{\mathbf{w}, \mathbf{f}\}$ wie folgt:

1. Für $p_i \in \text{ATOM}$ sei $\llbracket p_i \rrbracket \rho := \rho(p_i)$.
2. $\llbracket \perp \rrbracket \rho := \mathbf{f}$
3. Für Propositionen $A, B \in \text{PROP}$ seien:
 - (a) $\llbracket A \wedge B \rrbracket \rho := \llbracket A \rrbracket \rho \wedge \llbracket B \rrbracket \rho$
 - (b) $\llbracket A \vee B \rrbracket \rho := \llbracket A \rrbracket \rho \vee \llbracket B \rrbracket \rho$
 - (c) $\llbracket A \rightarrow B \rrbracket \rho := \llbracket A \rrbracket \rho \rightarrow \llbracket B \rrbracket \rho$

Bemerkung. Offensichtlich hängt $\llbracket A \rrbracket \rho$ nur von der Belegung der atomaren Aussagen ab, die tatsächlich in A vorkommen. In der Praxis gibt man deshalb oft nur die Werte von ρ an, die tatsächlich gebraucht werden.

Bemerkung. Für die Semantik der abgeleiteten Junktoren \neg und \leftrightarrow erhalten wir, dass \neg gerade durch $\bar{}$ und \leftrightarrow durch die boolesche Operation

\leftrightarrow	\mathbf{w}	\mathbf{f}
\mathbf{w}	\mathbf{w}	\mathbf{f}
\mathbf{f}	\mathbf{f}	\mathbf{w}

interpretiert wird.

3.2 Auswertungshomomorphismus

Wie wir bereits gesehen haben, können wir PROP als Termalgebra verstehen. In dieser Sichtweise ist $\llbracket \cdot \rrbracket \rho$ gerade der Boolesche-Algebren-Homomorphismus $T_{\Sigma}(\text{ATOM}) \rightarrow \{\mathbf{w}, \mathbf{f}\}$, den wir aus dem Fortsetzungssatz A.13.3 erhalten, weshalb die gleiche Bezeichnung $\llbracket \cdot \rrbracket \rho$ auch nicht zu Verwechslungen führen kann.

3.3 Gültigkeit und Erfüllbarkeit

Jetzt können wir auch erklären, was es heißt, dass eine Aussage gilt.

Definition 1.4. Sei $A \in \text{PROP}$ eine Proposition.

1. Für eine Belegung $\rho: \text{ATOM} \rightarrow \{\mathbf{w}, \mathbf{f}\}$ sagen wir, A **gilt unter der Belegung** ρ , wenn $\llbracket A \rrbracket_\rho = \mathbf{w}$.
2. Die Proposition A heißt **erfüllbar**, wenn es eine Belegung ρ mit $\llbracket A \rrbracket_\rho = \mathbf{w}$ gibt.
3. Wir sagen A ist eine **Tautologie** (oder **allgemeingültig**), oder auch A **gilt**, wenn A unter *jeder* Belegung ρ gilt. Wir schreiben dafür:

$$\vDash A$$

4. Wir nennen A **unerfüllbar** (**widersprüchlich** oder **kontradiktorisch**), wenn es keine Belegung ρ mit $\llbracket A \rrbracket_\rho = \mathbf{w}$ gibt, wenn also für alle Belegungen ρ immer $\llbracket A \rrbracket_\rho = \mathbf{f}$ gilt.

Da jede Proposition A nur endlich viele atomare Aussagen enthält, kann man diese Eigenschaften mit Hilfe einer **Wahrheitstafel** überprüfen.

3.4 Substitutionslemma

Gelegentlich ist folgende Variante von Lemma A.13.9 nützlich:

Lemma 1.5. Seien $B, B' \in \text{PROP}$ Propositionen, die $\vDash B \leftrightarrow B'$ erfüllen, $A \in \text{PROP}$ eine Proposition und $p \in \text{ATOM}$ eine atomare Aussage, dann gilt $\vDash A[B/p]$ genau dann wenn $\vDash A[B'/p]$ gilt, wobei $[B/p]$ die Substitution von p durch B bezeichne.

Die analoge Aussage gilt für Erfüllbarkeit bzw. Widersprüchlichkeit.

3.5 Semantische Konsequenz

Nun wollen wir formalisieren, was es bedeutet, dass eine Aussage die *Konsequenz* einer anderen, oder allgemeiner einer Menge von Aussagen ist.

Definition 1.6. Sei $\Gamma \subseteq \text{PROP}$ eine Menge von Propositionen und $A \in \text{PROP}$ eine Proposition. Dann nennen wir A eine **semantische Konsequenz** von Γ und schreiben dafür

$$\Gamma \vDash A,$$

wenn für jede Belegung $\rho: \text{ATOM} \rightarrow \{\mathbf{w}, \mathbf{f}\}$, unter der alle Propositionen aus Γ gelten (also $\llbracket C \rrbracket_\rho$ für alle $C \in \Gamma$), $\llbracket A \rrbracket_\rho = \mathbf{w}$ erfüllt ist, also A unter ρ gilt.

Wenn Γ endlich ist, können wir dies ebenfalls durch eine Wahrheitstafel überprüfen.