Logik

Arnd Poetzsch-Heffter

TU Kaiserslautern

SoSe 2017

Logik

Vorlesung:
Mi 11.45 - 13.15 Uhr   52-207

- Informationen
  https://softech.informatik.uni-kl.de/homepage/de/teaching/SS17/logik/
- Studienmaterial zur Vorlesung:
  - Folien (Dank an Prof. Madlener und Prof. Meyer!)
  - Handschriftliche Aufzeichnungen
  - Skript *Einführung in die Logik*
  - Uwe Schöning: Logik für Informatiker (Spektrum Akademischer Verlag)
  - Peter B. Andrews: An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof (Springer Science+Business Media)
  - Herbert B. Enderton: A Mathematical Introduction to Logic (second edition; Harcourt Academic Press)

Organisatorisches

Übungsblätter:

- Jede Woche ein Blatt, alternierend Präsenz- und Abgabeblatt
- Ausnahmsweise heute: erstes Präsenz- und Abgabeblatt
- Abgabe freitags, Postfach nahe Raum 401 und SoftTech AG
- Gruppen zu dritt

Übungen:

- Abwechselnd Präsenzübungen und Abgabeübungen
- Erste Übung in dieser Woche (Präsenzübung)
- Anmeldung im STAT-System (siehe Website; bis heute 18:00 Uhr)
- Tutoren: Lea Plückebaum, Albert Schimpf, Kathrin Thomas

Zulassungsvoraussetzungen für die Prüfung (Klausur):

- Teilnahme an den Übungen (Pflicht)
- Mindestens 50% der Punkte auf die abgegebenen Aufgaben
- Zwischenklausur bestanden (19.06.2017, 19:00 Uhr)

Inhalt

4. Grundlagen der Prädikatenlogik
   - Syntax
   - Semantik
   - Substitution
   - Normalformen
   - Herbrand-Theorie
   - Semi-Entscheidbarkeit der Allgemeingültigkeit
   - Untere Schranke für Allgemeingültigkeit
   - Kompaktheitssatz der Prädikatenlogik erster Stufe
5. Deduktive Systeme der Prädikatenlogik
   - Logische Folgerung
   - Das deduktive System $\mathcal{F}$
   - Theorien erster Stufe
   - Axiomatisierung
6. Algorithmische Verfahren für die Prädikatenlogik
   - Semantische Tableaus
   - Unifikation
   - Resolution

Einleitung

Zitate zur Logik

Mein teurer Freund, ich rat Euch drum
Zuerst **Collegium Logicum**.
Da wird der Geist Euch wohl dressiert,
In spanische Stiefeln eingeschnürt,
Daß er bedächtiger so fortan
Hinschleiche die Gedankenbahn,
Und nicht etwa, die Kreuz und Quer,
Irrlichteliere hin und her.
(aus J. W. v. Goehte, Faust I)

Die Wissenschaft, welche die Prinzipien des reinen Denkens enthält, wird transzendentale **Logik** genannt.
(aus I. Kant, Kritik der reinen Vernunft; in der Formulierung angepasst)

Logik in den Wissenschaften

- Philosophische Logik
- Logik in der Mathematik:
  - Grundlage mathematischer Beweise
  - Wie kann man Widersprüche vermeiden?
  - Lassen sich alle mathematischen Sätze beweisen?
- Logik in der Informatik:
  - Wie kann man Probleme und Software-Eigenschaften präzise beschreiben?
  - Wie kann man Probleme automatisch lösen oder Eigenschaften verifizieren?
  - Welche Beweise kann man automatisieren?

Beweise in Mathematik und Informatik

Theorem: Es gibt unendlich viele Primzahlen.

Beweis: (s. Vorlesung)

Theorem: Jede gerade Zahl größer 2 ist Summe zweier Primzahlen.

Beweis: ?

Theorem: Folgende Prozedur termiert für alle Eingaben größer Null:

```
    void collatz (int n) {
      if        n == 1     then  return
      else if   isEven(n)  then  collatz(n/2)
                           else  collatz(3*n+1)
}
```

Beweis: ?

Probleme beschreiben und lösen

Beispiel:

Von den fünf Kindern Albert, Bertold, Christian, Detlef und Emil lügen zwei immer, die anderen drei sagen immer die Wahrheit.

Wir erleben folgende Unterhaltung:
Albert: "Wenn Detlef die Wahrheit sagt, lügt Bertold."
Bertold: "Wenn Christian nicht lügt, dann ist entweder Albert oder Detlef ein Lügner."
Christian: "Emil lügt, und auch Albert oder Bertold lügen."
Detlef: "Wenn Bertold die Wahrheit sagt, dann auch Albert oder Christian."
Emil: "Unter den Personen Albert, Christian und Detlef befindet sich mindestens ein Lügner."

Fragen:

- Lässt sich eindeutig schlussfolgern, wer die Lügner sind?
- Wie kann man das Problem lösen?
- Kann man das Problem automatisch lösen?

Typische Problemstellungen der Informatik

- Spezifikation: Welche Eigenschaften soll mein Programm/meine Software genau haben? Z. B. bzgl. des Eingabe-Ausgabe-Verhaltens?
- Terminierung: Terminiert meine Software für alle Eingaben?
- Deadlock-Freiheit: Kann es sein, dass mein parallel arbeitendes Programm sich blockiert?
- Rückwärtskompatibilität: Kann ich meine neue Softwareversion überall dort einsetzen, wo ich die alte verwendet habe?
- Wissensverarbeitung: Wie kann man neues Wissen aus einer Wissenbasis ableiten? Wie kann man Wissen auf ein Problem anwenden?

Unterschiedliche Logiken

- **Aussagenlogik**
- **Prädikatenlogik** (erster Stufe)
- Logiken höherer Stufe, Typentheorie
- Programmlogiken (z.B. Hoare-Logik), dynamische Logiken
- Temporallogiken (LTL, CTL)
- Modallogiken, epistemische Logiken

Allgemeine Grundlagen formaler Logik

Eine **formale Logik** wird definiert durch:

- **Formale Sprache** für die Aussagen/Formeln mit festgelegter
  - **Syntax**: Welche Form haben die Aussagen?
  - **Semantik**: Was bedeuten die Aussagen? Welche Aussagen sind wahr?
- **Deduktives System/Deduktionssystem/Kalkül** für Beweise: Wie kann ich aus wahren Aussagen weitere ableiten/beweisen?

Wichtige Eigenschaften von Logiken

- **Ausdruckskraft**: Was kann man in der Logik ausdrücken?
- **Korrektheit**: Sind alle Aussagen, die man in der Logik beweisen kann, auch wahr?
- **Vollständigkeit**: Kann man alle wahren Aussagen der Logik mit dem deduktiven System der Logik beweisen?
- **Entscheidbarkeit**: Kann man automatisch berechnen, ob eine Aussage wahr ist?
- **Automatisierung**: Was kann man automatisch berechnen?

Kapitel I

Aussagenlogik

Die Sprache der Aussagenlogik

- Form/Aufbau von Aussagen $\rightsquigarrow$ **Syntax**

- Bedeutung von Aussagen $\rightsquigarrow$ **Semantik**: **wahr** (1), **falsch** (0)

Syntax der Aussagenlogik

Definition 1.1 (Syntax)

Betrachte das Alphabet $\Sigma = V \cup O \cup K$ mit

$V = \{p_1, p_2, ...\}$ einer abzählbaren Menge von **Aussagevariablen**,

$O = \{\neg/_1, \wedge/_2, \vee/_2, \rightarrow /_2, \leftrightarrow /_2\}$ **Verknüpfungen** mit Stelligkeiten (Operatoren, Junktoren),

$K = \{(,)\}$, **Klammern** (Hilfssymbole).

Die Menge der **Aussagen** oder **Formeln** der Aussagenlogik $F \subseteq \Sigma^*$ ist *induktiv definiert* durch:

1. $V \subseteq F$, die Menge der **atomaren Aussagen/Formeln**
2. Für $A, B \in F$ sind $(\neg A), (A \wedge B), (A \vee B), (A \rightarrow B), (A \leftrightarrow B) \in F$

d.h. F ist die *kleinste Menge*, die (1.) und (2.) erfüllt.

Strukturelle Induktion

Eigenschaften von Elementen in F werden durch **strukturelle Induktion**, d.h. durch Induktion über den Aufbau der Formeln, nachgewiesen.

Zum Beispiel sei $f : F \times \mathbb{N} \to \mathbb{N}$ definiert durch

$f(A, i) :=$ Anzahl an öffnenden Klammern minus Anzahl der schließenden Klammern in den ersten i Zeichen von A.

Folgende Lemmata lassen sich durch strukturelle Induktion zeigen:

Lemma 1.2

Für nicht-atomare Formeln $A \in F$ und für i, $1 \leq i < |A|$, gilt: $f(A, i) > 0$.
Für alle Formeln $B \in F$ gilt: $f(B, |B|) = 0$.

Korollar 1.3

Sei $A \in F$ und $B \in \Sigma^*$ ein echtes Präfix von A. Dann gilt $B \notin F$.

Satz 1.4 (Eindeutigkeitssatz)

Jede Formel $A \in F$ *ist entweder atomar oder lässt sich eindeutig darstellen als* $A \equiv (\neg A_1)$ *oder* $A \equiv (A_1 * A_2)$ *mit* $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ *und* $A_1, A_2 \in F$.

Dabei ist $\equiv\, \subseteq \Sigma^* \times \Sigma^*$ die syntaktische Gleichheit von Worten, die Formeln stimmen also Zeichen für Zeichen überein.

Vereinbarungen: Abkürzungen und Präzedenzregeln

- Beispiele für vollständig geschriebene aussagenlogische Formeln:

$$p_1, p_{101}, (((p_1 \rightarrow p_2) \wedge (\neg p_2)) \rightarrow (\neg p_1)), (p_1 \vee (\neg p_1))$$

- Zur besseren Lesbarkeit und kürzeren Schreibung:
  - Äußere Klammern können weggelassen werden.
  - Präzedenzregeln: $\neg$ bindet am stärksten, dann mit nachlassender Bindung $\wedge, \vee, \rightarrow, \leftrightarrow$
  - Ausdrücke mit mehreren $\wedge$- bzw. $\vee$-Operatoren werden links geklammert
  - Beispiele:

$$\begin{array}{rll} A \wedge B \rightarrow C & \text{steht für} & ((A \wedge B) \rightarrow C) \\ A \vee B \wedge C & \text{steht für} & (A \vee (B \wedge C)) \\ \neg A \vee B \wedge C & \text{steht für} & ((\neg A) \vee (B \wedge C)) \\ A \vee B \vee C & \text{steht für} & ((A \vee B) \vee C) \quad \text{(Linksklammerung)} \end{array}$$

Semantik

Semantik weist einer Aussage einen Wahrheitswert zu, also wahr (1) oder falsch (0). Sie "bewertet" also jede Aussage. Dafür muss man im Allgemeinen allerdings wissen, ob die Aussagenvariablen für wahre oder falsche Aussagen stehen.

Definition 1.5 (Variablenbelegung)

Eine **Belegung** der Variablen V ist eine Funktion $\psi : V \rightarrow \mathbb{B}$

Definition 1.6 (Bewertung)

Sei ψ eine Variablenbelegung. Dann heißt die Funktion $\mathcal{B}_\psi : F \to \mathbb{B}$ die **Bewertung** (der Formeln) unter ψ, wobei $\mathcal{B}_\psi$ wie folgt definiert ist:

$$\begin{aligned}
\mathcal{B}_\psi(p_i) &= \psi(p_i) \\
\mathcal{B}_\psi(\neg A) &= 1 - \mathcal{B}_\psi(A) \\
\mathcal{B}_\psi(A \vee B) &= max(\mathcal{B}_\psi(A), \mathcal{B}_\psi(B)) \\
\mathcal{B}_\psi(A \wedge B) &= min(\mathcal{B}_\psi(A), \mathcal{B}_\psi(B)) \\
\mathcal{B}_\psi(A \to B) &= \begin{cases} 0 & \text{falls } \mathcal{B}_\psi(A) = 1 \text{ und } \mathcal{B}_\psi(B) = 0 \\ 1 & \text{sonst} \end{cases} \\
\mathcal{B}_\psi(A \leftrightarrow B) &= \begin{cases} 0 & \text{falls } \mathcal{B}_\psi(A) \neq \mathcal{B}_\psi(B) \\ 1 & \text{falls } \mathcal{B}_\psi(A) = \mathcal{B}_\psi(B) \end{cases}
\end{aligned}$$

Belegungen und Bewertungen (Fort.)

- Sprechweise: A ist **falsch** unter ψ, falls $\mathcal{B}_\psi(A) = 0$
  A ist **wahr** unter ψ oder ψ **erfüllt** A, falls $\mathcal{B}_\psi(A) = 1$.
- Man kann $\mathcal{B}$ auch als Funktion höherer Ordnung verstehen, die als erstes Argument die Belegung nimmt:
  $\mathcal{B} : (V \to \mathbb{B}) \to F \to \mathbb{B}$
- Darstellung von Bewertungen durch **Wahrheitstafeln**:

p	$\neg p$
1	0
0	1

p	q	$p \vee q$	$p \wedge q$	$p \to q$	$p \leftrightarrow q$
0	0	0	0	1	1
0	1	1	0	1	0
1	0	1	0	0	0
1	1	1	1	1	1

Belegungen und Bewertungen (Fort.)

Lemma 1.7

Die Bewertung einer Formel $A \in F$ hängt nur von der Belegung der in ihr vorkommenden Aussagevariablen aus V ab. Das heißt, will man $\mathcal{B}_\psi(A)$ berechnen, genügt es, die Werte $\psi(p)$ für alle in A vorkommenden Aussagevariablen p zu kennen.

- Beispiel: Sei $\psi(p) = 1, \psi(q) = 1, \psi(r) = 0$. Dann kann $\mathcal{B}_\psi(A)$ iterativ berechnet werden:

$$A \equiv \underbrace{\underbrace{(\underbrace{p}_{1} \to \underbrace{(\underbrace{q}_{1} \to \underbrace{r}_{0})}_{0})}_{0} \to \underbrace{(\underbrace{(\underbrace{p}_{1} \land \underbrace{q}_{1})}_{1} \to \underbrace{r}_{0})}_{0}}_{1}$$

Also gilt $\mathcal{B}_\psi(A) = 1$.

Belegungen und Bewertungen (Fort.)

Seien $V_A = \{p_1, ..., p_n\}$ die Variablen in A. Welche Werte nimmt $\mathcal{B}_\psi(A)$ an, wenn ψ **alle** Belegungen durchläuft?

- Ist etwa $\mathcal{B}_\psi(A) = 1$ für alle Belegungen ψ?
- Es genügt, die **endlich** vielen Belegungen zu prüfen, die sich auf den Variablen in V_A unterscheiden (2^n verschiedene).
- Jede solche Belegung läßt sich als ein Vektor aus $\mathbb{B}^n$ darstellen.

A definiert eine boolesche Funktion $f_A : \mathbb{B}^n \to \mathbb{B}$.

- Beispiel: Für die drei Variablen p, q und r aus A im obigen Beispiel gibt es 8 Belegungen, die betrachtet werden müssen.

Belegungen und Bewertungen (Fort.)

p	q	r	$q \to r$	$p \wedge q$	$p \to (q \to r)$	$(p \wedge q) \to r$	A
0	0	0	1	0	1	1	1
0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1
0	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1
1	0	1	1	0	1	1	1
1	1	0	0	1	0	0	1
1	1	1	1	1	1	1	1

A ist wahr unabhängig von den Werten von p, q, r, d.h. für jede Bewertung $\mathcal{B}_\psi$. Weitere solche Formeln sind etwa: $(A \to (B \to A))$, $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$ oder $((\neg A \to \neg B) \to (B \to A))$.

Erfüllbarkeit und weitere wichtige Begriffe

Definition 1.8

Seien $A \in F$, $\Sigma \subseteq F$.

(a) A heißt **Tautologie** (**allgemeingültig**), falls $\mathcal{B}_\psi(A) = 1$ für jede Belegung ψ gilt. (Schreibweise $\models A$)

(b) A ist **erfüllbar**, falls es eine Belegung ψ gibt mit $\mathcal{B}_\psi(A) = 1$.

(c) Σ ist **erfüllbar**, falls es eine Belegung ψ gibt mit $\mathcal{B}_\psi(A) = 1$ für alle $A \in \Sigma$. (ψ **erfüllt** Σ)

(d) A ist **widerspruchsvoll**, falls $\mathcal{B}_\psi(A) = 0$ für jede Belegung ψ.

(e) TAUT $:= \{A \in F \mid A \text{ ist Tautologie}\}$ die **Menge der Tautologien**.

(f) SAT $:= \{A \in F \mid A \text{ ist erfüllbar}\}$ die **Menge der erfüllbaren Formeln**.

Beachte TAUT $\subseteq$ SAT.

Semantische Folgerung

Definition 1.9

Seien $A \in F$, $\Sigma \subseteq F$.

(a) **Semantischer Folgerungsbegriff:** A ist **logische Folgerung** von Σ, falls für jede Belegung ψ, die Σ erfüllt, $\mathcal{B}_\psi(A) = 1$ gilt.

Man schreibt $\Sigma \models A$. Auch $A_1, \ldots, A_n \models A$, falls $\Sigma = \{A_1, \ldots, A_n\}$.

(b) Die Menge $\text{Folg}(\Sigma)$ der Folgerungen aus Σ ist definiert durch:

$$\text{Folg}(\Sigma) := \{A \in F \mid \Sigma \models A\}.$$

Beispiele

Beispiel 1.10

1. $(p \vee (\neg p))$, $((p \to q) \vee (q \to r))$, $p \to (q \to p)$, $(p \to p)$, $(p \to \neg\neg p)$ und A aus dem Beispiel nach Lemma 1.7 sind Tautologien.
2. $(p \wedge (\neg p))$ ist widerspruchsvoll.
3. $(p \wedge q)$ ist erfüllbar, aber weder Tautologie noch widerspruchsvoll.
4. Sei $\Sigma = \{p\}$ und $A = p \vee q$. Dann gilt $\Sigma \models A$, denn falls $\psi(p) = 1$, dann auch $\mathcal{B}_\psi(p \vee q) = 1$. Jede Belegung, die Σ erfüllt, erfüllt also auch A.

Einfache Eigenschaften

Lemma 1.11

(a) *A allgemeingültig gdw. $\neg A$ widerspruchsvoll.*

(b) *Es gilt $\emptyset \models A$ genau dann, wenn A Tautologie ist: $Folg(\emptyset) = TAUT$.*

(c) *Ist Σ nicht erfüllbar, dann gilt $\Sigma \models A$ für alle $A \in F$: $Folg(\Sigma) = F$. Insbesondere $\Sigma \models A$ und $\Sigma \models \neg A$ für ein $A \in F$.*

(d) *Sei $\Sigma \subseteq \Sigma'$. Ist Σ' erfüllbar, dann ist auch Σ erfüllbar.*

(e) *Es gilt $\Sigma \subseteq Folg(\Sigma)$ und $Folg(Folg(\Sigma)) = Folg(\Sigma)$.*

(f) *Falls $\Sigma \subseteq \Sigma'$, dann gilt $Folg(\Sigma) \subseteq Folg(\Sigma')$.*

(g) *$\Sigma \models A$ gilt genau dann, wenn $\Sigma \cup \{\neg A\}$ nicht erfüllbar.*

(h) *Ist Σ endlich, dann ist es entscheidbar, ob Σ erfüllbar ist, und die Menge $Folg(\Sigma)$ ist entscheidbar.*

(i) *Die Mengen TAUT und SAT sind entscheidbar.*

Semantische Versionen vom Deduktionstheorem und Modus-Ponens

Lemma 1.12

a) **Deduktionstheorem** *(semantische Version):*

$$\Sigma, A \models B \qquad \text{gdw} \qquad \Sigma \models (A \rightarrow B).$$

(Σ, A ist Kurzschreibweise für $\Sigma \cup \{A\}$)

b) **Modus-Ponens** *(semantische Version):*

$$\{A, A \rightarrow B\} \models B.$$

Insbesondere ist B Tautologie, falls A und $(A \rightarrow B)$ Tautologien.

Logische Äquivalenz

Definition 1.13 (Logische Äquivalenz)

Formeln $A, B \in F$ heißen **logisch äquivalent**, $A \models\mid B$, falls für jede Belegung ψ gilt: $\mathcal{B}_\psi(A) = \mathcal{B}_\psi(B)$.

Beispiele logisch äquivalenter Formeln:

(Involution)	$A \models\mid \neg(\neg A)$	
(Idempotenz)	$A \models\mid A \wedge A$	$A \models\mid A \vee A$
(Kommutativität)	$A \wedge B \models\mid B \wedge A$	$A \vee B \models\mid B \vee A$
(Assoziativität)	$A \wedge (B \wedge C) \models\mid (A \wedge B) \wedge C$	
	$A \vee (B \vee C) \models\mid (A \vee B) \vee C$	
(Distributivität)	$A \wedge (B \vee C) \models\mid (A \wedge B) \vee (A \wedge C)$	
	$A \vee (B \wedge C) \models\mid (A \vee B) \wedge (A \vee C)$	

Logische Äquivalenz (Fort.)

(De Morgan)	$\neg(A \wedge B) \models\mid \neg A \vee \neg B$	$\neg(A \vee B) \models\mid \neg A \wedge \neg B$
	$A \rightarrow B \models\mid \neg A \vee B$	$A \leftrightarrow B \models\mid (A \rightarrow B) \wedge (B \rightarrow A)$
	$A \vee B \models\mid \neg A \rightarrow B$	$A \wedge B \models\mid \neg(A \rightarrow \neg B)$

Lemma 1.14

Logische Äquivalenz $\models\mid \subseteq F \times F$ *ist eine Äquivalenzrelation, also reflexiv, symmetrisch und transitiv.*

Es gilt sogar: Ersetzt man in einer Formel A *eine Teilformel* B *durch* C *mit* $C \models\mid B$*, so gilt:* $A \models\mid A[C/B]$.

Logische Äquivalenz (Fort.)

Lemma 1.15

Folgende Aussagen sind äquivalent:

$$\models A \leftrightarrow B \qquad\qquad A \models\!\!\!\dashv B$$

$$A \models B \text{ und } B \models A \qquad\qquad Folg(A) = Folg(B)$$

Lemma 1.16

Zu jeder Formel $A \in F$ gibt es $B, C, D \in F$ mit

1. $A \models\!\!\!\dashv B$, *B enthält nur $\rightarrow$ und $\neg$ als Verknüpfungen*
2. $A \models\!\!\!\dashv C$, *C enthält nur $\wedge$ und $\neg$ als Verknüpfungen*
3. $A \models\!\!\!\dashv D$, *D enthält nur $\vee$ und $\neg$ als Verknüpfungen*

Folgt aus obigen Äquivalenzen.

Varianten der Sprache der Aussagenlogik

Man kann die Aussagenlogik auch über Formeln mit weniger oder mehr Operatoren definieren. Sei OP eine Menge von Operatoren, dann bezeichnen wir mit F_{OP} die Formeln, in denen nur Operatoren aus OP verwendet werden. Es gilt also $F = F_{\{\neg,\vee,\wedge,\rightarrow,\leftrightarrow\}}$.

Definition 1.17 (Vollständige Operatorenmengen)

Eine Menge OP von Operatoren heißt **vollständig**, falls es zu jedem $A \in F$ eine logisch äquivalente Formel $B \in F_{OP}$ gibt.

- Vollständige Operatorenmengen für die Aussagenlogik sind z.B.: $\{\neg, \rightarrow\}, \{\neg, \vee\}, \{\neg, \wedge\}, \{\neg, \vee, \wedge\}, \{\text{false}, \rightarrow\}$

  Dabei ist false eine Konstante mit $\mathcal{B}_\psi(\text{false}) = 0$ für jedes ψ. Offenbar gilt: $\neg A \models\!\!\!=\!\!\!\!\mid (A \rightarrow \text{false})$.
- **Normalformen**: DNF (Disjunktive Normalform), KNF (Konjunktive Normalform), KDNF, KKNF (Kanonische Formen).

Boolsche Funktionen

Jede Formel A mit den Variablen $V_A = \{p_1, ..., p_n\}$ stellt eine boolsche Funktion $f_A : \mathbb{B}^n \to \mathbb{B}$ dar, nämlich mit der Definition $f_A(b_1, ..., b_n) := \mathcal{B}_\psi(A)$, wobei $\psi(p_i) := b_i$.

- Man kann zeigen, dass sich jede boolsche Funktion $f : \mathbb{B}^n \to \mathbb{B}$ $(n > 0)$ in obiger Form durch eine Formel A mit vollständiger Operatormenge darstellen lässt.
- Die boolsche Algebra hat als übliche Operatormenge **true, false, not, or, and**.
- Für andere Operatormengen, die etwa **nand, nor** enthalten, siehe Digitale Logik. Dort werden nand, nor-Gatter bevorzugt, da sie nur zwei Transistoren benötigen.

Boolsche Funktionen: Beispiel

Ein Patientenüberwachungssystem erhält gewisse Daten über den Zustand eines Patienten: Temperatur, Blutdruck, Pulsrate. Die Schwellenwerte für die Daten seien wie folgt festgelegt:

Zustände Ein/Ausgaben	Bedeutung
A	Temperatur außerhalb 36-39°C.
B	Blutdruck außerhalb 80-160 mm.
C	Pulsrate außerhalb 60-120 Schläge pro Minute.
O	Alarmaktivierung ist notwendig.

Boolsche Funktionen: Beispiel (Fort.)

Die Anforderungen, d.h. bei welchen Kombinationen der Werte der Zustände eine Alarmaktivierung notwendig ist, werden durch den Medizin-Experten festgelegt. Sie seien in folgender Tabelle fixiert:

I/*O* - **Tabelle**

A	B	C	O
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Logischer Entwurf: Betrachte die Zeilen in denen O den Wert 1 hat und stelle eine DNF auf:

$$(\neg A \wedge B \wedge C) \vee (A \wedge \neg B \wedge C) \vee$$
$$(A \wedge B \wedge \neg C) \vee (A \wedge B \wedge C)$$

Boolsche Funktionen: Beispiel (Fort.)

Als eine Realisierung könnte man das folgende Schaltnetz nehmen: