

1. Informatik: Eine Übersicht

2. Logik (Einführung / Grundlagen)

2.1 Aussagenlogik

2.2 Prädikatenlogik

2.3 Boolesche Algebra

3. Objektorientierte Modellierung

4. Algorithmen und ihre Eigenschaften

5. Entwurfsmethoden von Algorithmen

6. Sortieralgorithmen

7. Dynamische Datenstrukturen



Warum Logik in der Angewandten Informatik ?

- Einübung logischen Denkens
- Formale Spezifikation von Anwenderproblemen
- Semantik von Programmiersprachen
- Programmverifikation
- Logische Programmierung
- Wissensrepräsentation
- logische Gatter → siehe Grundlagen II
- Semantic Web

■ Beispiel 2.0

Auto- und Motorradvermietung

Zu vermieten:

„Mercedes-Benz SL 280“ (M1), „Porsche 996 C2“ (P1),
„Honda Accord“ (H1), „BMW R 1200 C 0“ (B1), „Suzuki AN 400“ (S1)



Faktenbasis (5 Instanzen):

cabrio(M1), cabrio(P1), auto(H1), motorrad(B1), motorrad(S1)

Regelbasis (3 Axiome):

$\forall x \text{ cabrio}(x) \Leftrightarrow \text{auto}(x) \wedge \text{offen}(x)$

$\forall x \text{ auto}(x) \Rightarrow \text{fahrzeug}(x) \wedge \text{vierräder}(x)$

$\forall x \text{ motorrad}(x) \Leftrightarrow \text{offen}(x) \wedge \text{fahrzeug}(x) \wedge \text{zweiräder}(x)$

Vorgriff: Beispiel ist aus der Prädikatenlogik

Anfragen:

fahrzeug(S1) = wahr

auto(S1) = falsch

Kompakte Darstellung und Nutzung von Wissen

5 Instanzen + 3 Regeln →

30 beantwortbare Fragen:

- Ist S1 ein Fahrzeug?
- Ist S1 ein Auto?

?x: fahrzeug(x)



?x: auto(x)



?x: offen(x)



Beispiele für **atomare Aussagen**:

- „Der Mond kreist um die Erde“
- „Die Sonne kreist um die Erde“
- „17 ist eine Primzahl“
- „17 ist keine Primzahl“
- „ $14 + 6 = 6 + 14$ “
- „ $P = NP$ “

Keine atomaren Aussagen:

- „Bayer 04 wird nächstes Jahr deutscher Fußball-Meister“
- „ $x = 17$ “
- „diese Aussage ist falsch“

Jede atomare Aussage hat einen **Wahrheitswert**:

(„wahr“, „falsch“), entspricht („true“, „false“), entspricht (1,0)

Atomare Aussagen lassen sich

- nicht weiter in Bestandteile zerlegen
- zu Formeln zusammensetzen

"Der Mond kreist um die Erde."
"Die Sonne kreist um die Erde."

"Der Mond kreist um die Erde
und die Sonne kreist um die
Erde."

Zu unterscheiden von
der Aussage:

Aussagenlogik behandelt die Struktur syntaktisch zusammengesetzter Formeln abstrakter Aussagen:

$(A1 \wedge A2), ((\neg A1 \vee A2) \wedge A3), \dots$

$A1, A2, A3$ Aussagezeichen mit beliebigem Wahrheitswert

\wedge, \vee, \neg Symbole für logische Kombinationen

Definition: Syntax der Aussagenlogik

Geg: Menge von Aussagezeichen $AZ = \{A1, A2, \dots\}$

Die Menge **AL** aller **aussagenlogischen Formeln** ist induktiv definiert:

- (i) Jedes A aus AZ ist in AL.
- (ii) Ist F in AL, so ist auch $\neg F$ in AL
- (iii) Sind F und G in AL, so ist auch $(F \wedge G)$ in AL.
- (iv) Sind F und G in AL, so ist auch $(F \vee G)$ in AL.
- (v) Induktive Definition:
Alle Elemente aus AL sind durch (i) - (iv) entstanden!

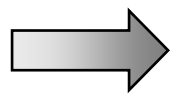
Bezeichnungen:

A_i	atomare Formel
$\neg F$	Negation (von F)
$(F \wedge G)$	Konjunktion (von F und G)
$(F \vee G)$	Disjunktion (von F und G)

z. B. $((F \vee G \wedge (G \vee \neg F)) \notin AL$

$((F \vee G) \wedge (G \vee \neg F)) \in AL$

Um eine **Eigenschaft P** für alle Formeln **F** aus **AL** zu beweisen, kann man einen Beweis durch Induktion führen:



strukturelle Induktion bzw.
Induktion über den Formelaufbau

Induktionsanfang:

1. P gilt für jedes A aus AZ

Induktionsschritt:

2. Wenn P für F gilt, dann auch für $\neg F$.
3. Wenn P für F und G gilt, dann auch für $(F \wedge G)$
4. Wenn P für F und G gilt, dann auch für $(F \vee G)$

Induktionsannahme ist dabei die Gültigkeit von P für F bzw. F und G.

■ Beispiel 2.1 **Induktionsbeweis**

P: In jeder Formel ist die Zahl der öffnenden Klammern gleich der Zahl der schließenden Klammern

Induktionsanfang:

- P gilt für jedes A aus AZ.
(ist die Formel atomar, besitzt sie keine Klammern;
es gilt die Behauptung)

Induktionsschritt:

- Wenn P für F gilt, dann auch für $\neg F$.
- Wenn P für F und G gilt, dann auch für $(F \wedge G)$ und $(F \vee G)$.
(bei jeder Anwendung einer der Regeln werden genauso viele öffnende wie schließende Klammern erzeugt: genau je eine!)

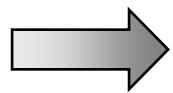
Induktionsannahme ist dabei die Gültigkeit von P
(also Anzahl der öffnenden Klammern = Anzahl der schließenden Klammern) für F bzw. F und G.

Definition: Semantik der Aussagenlogik

Die Elemente der Menge {falsch, wahr} heißen **Wahrheitswerte**.

(Es kann auch die Menge {0,1} oder die Menge {false, true} benutzt werden)

Eine **Belegung a** ist eine Funktion $a: AZ \rightarrow \{\text{falsch, wahr}\}$.



*Definition der Semantik der Zeichen " \neg ", " \wedge " und " \vee " durch Verwendung der Begriffe "**nicht**", "**und**", "**oder**".*

Für jede Belegung a sei a^* die eindeutig definierte Funktion
 $a^*: AL \rightarrow \{\text{falsch, wahr}\}$ mit:

(i) $a^*(A) = a(A)$ für alle A aus AZ
 (a ist die *Einschränkung* von a^* auf die Teilmenge $AZ \subseteq AL$,
 d.h. a^* ist die Erweiterung von a auf AL)

(ii) $a^*(\neg F) = \begin{cases} \text{wahr, falls } a^*(F) = \text{falsch} \\ \text{falsch, sonst} \end{cases}$

(iii) $a^*(F \wedge G) = \begin{cases} \text{wahr, falls } a^*(F) = \text{wahr und } a^*(G) = \text{wahr} \\ \text{falsch, sonst} \end{cases}$

(iv) $a^*(F \vee G) = \begin{cases} \text{wahr, falls } a^*(F) = \text{wahr oder } a^*(G) = \text{wahr} \\ \text{(oder beides)} \\ \text{falsch, sonst} \end{cases}$



Darstellung der Wirkungen der Operationen mit **Wahrheitstafeln**:

$a^*(F)$	$a^*(G)$	$a^*(F \wedge G)$	$a^*(F \vee G)$	$a^*(\neg F)$
falsch	falsch	falsch	falsch	wahr
falsch	wahr	falsch	wahr	wahr
wahr	falsch	falsch	wahr	falsch
wahr	wahr	wahr	wahr	falsch

Oder kürzer:

$a^*(F)$	$a^*(G)$	$a^*(F \wedge G)$	$a^*(F \vee G)$	$a^*(\neg F)$
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0



■ Beispiel 2.2

Geg.: $a(A) = \text{wahr}, a(B) = \text{wahr}, a(C) = \text{falsch}$

Dann lässt sich aus der Belegung der einzelnen Aussagezeichen der Formel schließen:

$$a^* ((A \vee B) \wedge \neg C) = \text{wahr},$$

denn: $a^* (A \vee B) = \text{wahr}$

$$a^* (\neg C) = \text{wahr}$$

In der Wahrheitstafel:

$a(A)$	$a(B)$	$a(C)$	$a^*(A \vee B)$	$a^*(\neg C)$	$a^*((A \vee B) \wedge \neg C)$
1	1	0	1	1	1



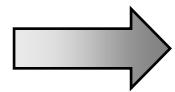
Entscheidend ist nur die Belegung der verwendeten Aussagezeichen:

Satz: (Koinzidenzatz der Aussagenlogik)

Seien a_1, a_2 Belegungen, die auf allen Aussagezeichen einer Formel F übereinstimmen.

Dann gilt: $a_1^*(F) = a_2^*(F)$,

d.h. $a_1^*(F)$ ist genau dann wahr, wenn $a_2^*(F)$ wahr ist.



Beweis: Durch strukturelle Induktion!



Definition:

Sei F eine aussagenlogische Formel und a eine Belegung.

Dann heißt:

- F **gültig** unter Belegung a , wenn $a^*(F) = \text{wahr}$
Man sagt: a **erfüllt** F oder a ist **Modell** von F
Schreibweise: $a \models F$

- b) F **erfüllbar**, wenn mind. eine Belegung a existiert mit $a \models F$
 F **unerfüllbar**, sonst.

- c) F heißt **allgemeingültig** (Tautologie), wenn $a \models F$ für alle Belegungen a gilt.
Schreibweise: $\models F$



■ Beispiel 2.3

• $a \models (A \wedge \neg B)$

gilt in der Belegung $a(A) = \text{wahr}$
und $a(B) = \text{falsch}$

Die Formel ist also

In der Wahrheitstafel:

$a(A)$	$a(B)$	$a^*(\neg B)$	$a^*(A \wedge \neg B)$
0	0		
0	1		
1	0		
1	1		

• $(A \wedge \neg A)$

ist

In der Wahrheitstafel:

$a(A)$	$a^*(\neg A)$	$a^*(A \wedge \neg A) = a^*(0)$
0	1	
1	0	

- $(\neg A \vee A)$ ist eine **Tautologie**.

Diese Formel ist also allgemeingültig.

In der Wahrheitstafel:

$a(A)$	$a(\neg A)$	$a^*(A \vee \neg A) = a^*(1)$
0	1	1
1	0	1

Bemerkung:

F allgemeingültig	\Leftrightarrow	\neg F unerfüllbar
F unerfüllbar	\Leftrightarrow	\neg F allgemeingültig



Definition:

Sei M eine Menge von Formeln, a eine Belegung und F eine Formel.

- a heißt **Modell** von M , wenn $a \models F$ für jede Formel F aus M
- M heißt **erfüllbar**, wenn es mindestens ein Modell für M gibt. Sonst heißt M **widerspruchsvoll**.
- F heißt (logische) **Folgerung** von M , wenn $a \models F$ für jedes Modell a von M

Schreibweise: $M \models F$



■ Beispiel 2.4

- $M = \{A, \neg A\}$ ist **widerspruchsvoll**,

denn für ein Modell a müsste gleichzeitig gelten:

$a \models A$ und $a \models \neg A$, d.h. $a(A) = \text{wahr}$ und $a(A) = \text{falsch}$

- **"Modus Ponens" - logische Folgerung**

Sei $M = \{A, (\neg A \vee B)\}$.

Dann gilt $M \models B$, denn für jedes Modell a von M gilt:

$a \models A$ und $a \models (\neg A \vee B)$,

d.h. $(a(A) = \text{wahr})$

und $(a(\neg A) = \text{wahr} \text{ oder } a(B) = \text{wahr})$.

Folglich kommt nur $a(B) = \text{wahr}$ in Frage,

somit: a **erfüllt** B .



Bemerkungen:

Deduktionstheorem:

$$M \models F \Leftrightarrow M \cup \{\neg F\} \text{ widerspruchsvoll}$$

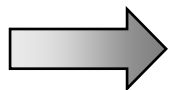
$$M \text{ widerspruchsvoll} \Leftrightarrow M \models F \text{ für alle } F \in AL$$

(da es kein Modell für M gibt, ist offensichtlich jedes Modell für M auch ein Modell für F !)

für eine endliche Menge $M = \{F_1, \dots, F_n\}$:

$$a \text{ erfüllt } (F_1 \wedge \dots \wedge F_n) \Leftrightarrow a \text{ Modell von } \{F_1, \dots, F_n\}$$

$$\begin{aligned} \{F_1, \dots, F_n\} \models F &\Leftrightarrow \{F_1, \dots, F_n, \neg F\} \text{ widerspruchsvoll} \\ &\Leftrightarrow (F_1 \wedge \dots \wedge F_n \wedge \neg F) \text{ unerfüllbar} \end{aligned}$$



Logische Folgerung für endliche Formelmengen wird auf den Begriff der Erfüllbarkeit zurückgeführt.

Man beachte:

Wahrheitstafeln ermöglichen die Überprüfung aller Belegungen einer Formel.

Es existieren folglich Algorithmen, um Allgemeingültigkeit und Erfüllbarkeit zu untersuchen.

F ist Tautologie	\Leftrightarrow	Wahrheitstafel liefert nur wahr (also allgemeingültig)
F erfüllbar	\Leftrightarrow	Wahrheitstafel liefert mindestens ein wahr
F unerfüllbar	\Leftrightarrow	Wahrheitstafel liefert kein wahr



■ Beispiel 2.5 Tautologie

Ist $((A \wedge B) \vee C) \vee (\neg(A \wedge B) \wedge \neg C)$ eine Tautologie?

Festlegung: $F1 = (A \wedge B)$ $F2 = \neg C$ $F3 = (F1 \vee C)$
 $F4 = (\neg F1 \wedge F2)$ $F5 = (F3 \vee F4)$


a(A)	a(B)	a(C)	a*(F1)	a*(F2)	a*(F3)	a*(F4)	a*(F5)
1	1	1	1	0	1	0	1
1	1	0	1	1	1	0	1
1	0	1	0	0	1	0	1
1	0	0	0	1	0	1	1
0	1	1	0	0	1	0	1
0	1	0	0	1	0	1	1
0	0	1	0	0	1	0	1
0	0	0	0	1	0	1	1

Wie aufwendig ist der Test auf Erfüllbarkeit?

Eine Formel F aus AL enthalte n Aussagezeichen A_1, \dots, A_n .

Aufwand für Bestimmung von $a^*(F)$ für **eine** Belegung $a: AZ \rightarrow \{\text{falsch, wahr}\}$ ist **linear in der Länge der Formel**, d.h. Anzahl der Aussagezeichen.

Maximal 2^n verschiedene Belegungen müssen untersucht werden.

 *Folglich entsteht im schlechtesten Fall **exponentieller Aufwand !!***



Gibt es ein Verfahren, das für jedes $F \in AL$ die Erfüllbarkeit testet und weniger als exponentiellen (also polynomiellen) Aufwand erfordert?

➔ Dies ist das **berühmteste offene Problem der Informatik.**

Eine positive Antwort auf diese Frage würde zu effizienten Lösungsverfahren für eine Vielzahl schwerer Probleme führen, insbesondere für fast alle im **Operations Research** betrachteten Optimierungsprobleme.



(siehe später Eigenschaften von Algorithmen)

Definition:

Seien F, G aussagenlogische Formeln.

- G ist **Implikation** von F , bzw. F **impliziert** G (in Zeichen: $(F \Rightarrow G)$), wenn $\{F\} \models G$.
- F, G sind **äquivalent** (in Zeichen: $(F \equiv G)$ oder $(F \Leftrightarrow G)$), wenn $(F \Rightarrow G)$ und $(G \Rightarrow F)$.

Lemma: $(F \Rightarrow G)$ genau dann, wenn $\models (\neg F \vee G)$

Beweis: über Wahrheitstafel



Satz: Ersetzbarkeit äquivalenter Formeln

Seien $F, G, H, H' \in AL$ mit:

$$F \equiv G;$$

in H kommt (mindestens einmal) F als Teilformel vor;

H' entsteht aus H , indem (irgend) ein Vorkommen von F in H durch G ersetzt wird.

Dann gilt: $H \equiv H'$.

Beweis: Durch strukturelle Induktion über den Formelaufbau von H .



Satz:

Es gelten folgende Äquivalenzen für beliebige Formeln F , G , H aus AL (Beweis über Wahrheitstafeln)

Idempotenz

$$(F \wedge F) \equiv F$$

$$(F \vee F) \equiv F$$

Kommutativität

$$(F \wedge G) \equiv (G \wedge F)$$

$$(F \vee G) \equiv (G \vee F)$$

Assoziativität

$$((F \wedge G) \wedge H) \equiv (F \wedge (G \wedge H))$$

$$((F \vee G) \vee H) \equiv (F \vee (G \vee H))$$

Distributivität

$$(F \wedge (G \vee H)) \equiv ((F \wedge G) \vee (F \wedge H))$$

$$(F \vee (G \wedge H)) \equiv ((F \vee G) \wedge (F \vee H))$$

Absorption

$$(F \wedge (F \vee G)) \equiv F$$

$$(F \vee (F \wedge G)) \equiv F$$

Doppelnegation

$$\neg \neg F \equiv F$$

(„Satz vom ausgeschlossenen Dritten“)

deMorgansche Regel $\neg(F \wedge G) \equiv (\neg F \vee \neg G)$
 $\neg(F \vee G) \equiv (\neg F \wedge \neg G)$

neutrale Elemente $((F \vee \neg F) \wedge G) \equiv G$
 $((F \wedge \neg F) \vee G) \equiv G$



■ Beispiel 2.6

Beweis der 1. deMorganschen Regel über Wahrheitstafel
(z.z. $\neg(F \wedge G) \equiv (\neg F \vee \neg G)$)

F	G	$\neg(F \wedge G)$	$\neg F$	$\neg G$	$(\neg F \vee \neg G)$
0	0	1	1	1	1
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	0	0	0

Wegen der Assoziativität können Klammern bei aufeinander folgenden " \wedge " oder " \vee " im weiteren weggelassen werden.



Definition: (Normalformen)

- Ein **Literal** L ist eine Formel der Form A_i (**positives Literal**) oder $\neg A_i$ (**negatives Literal**)

- Eine Formel F ist in **konjunktiver Normalform (KNF)**, wenn sie eine Konjunktion von Disjunktionen von Literalen ist:

$$(L_{1,1} \vee \dots \vee L_{1,n_1}) \wedge \dots \wedge (L_{j,1} \vee \dots \vee L_{j,n_j}) \quad n_i \in \mathbb{IN}$$

wobei $L_{i,k}$ positive oder negative Literale sind

- Eine Formel F ist in **disjunktiver Normalform (DNF)**, falls sie eine Disjunktion von Konjunktionen von Literalen ist:

$$(L_{1,1} \wedge \dots \wedge L_{1,n_1}) \vee \dots \vee (L_{j,1} \wedge \dots \wedge L_{j,n_j}) \quad n_i \in \mathbb{IN}$$

Satz:

Zu jeder Formel F existiert eine äquivalente Formel K in konjunktiver Normalform und eine äquivalente Formel D in disjunktiver Normalform. (K und D sind nicht eindeutig!)

Beweis:

- durch Induktion über den Formelaufbau von F oder
- durch Angabe korrekter konstruktiver Verfahren zur Herstellung von KNF / DNF



Zwei Vorgehensweisen zur Bestimmung von KNF und DNF:

Gegeben: Formel F

Gesucht: Eine zu F äquivalente Formel in KNF bzw. DNF

Algorithmus I: Bestimmung einer KNF

Schritt a)

- Negationszeichen von außen nach innen ziehen:

Ersetze in F jedes Vorkommen einer Teilformel der Bauart

$\neg \neg G$	durch	G
$\neg(G \wedge H)$	durch	$(\neg G \vee \neg H)$
$\neg(G \vee H)$	durch	$(\neg G \wedge \neg H)$

bis keine derartige Teilformel mehr vorkommt.

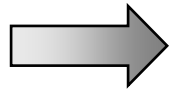
Schritt b)

- Alle " \vee "-Zeichen soweit wie möglich nach innen ziehen:

Ersetze jedes Vorkommen einer Teilformel der Bauart

$$\begin{array}{ll} (F \vee (G \wedge H)) & \text{durch } ((F \vee G) \wedge (F \vee H)) \\ ((F \wedge G) \vee H) & \text{durch } ((F \vee H) \wedge (G \vee H)) \end{array}$$

bis keine derartige Teilformel mehr vorkommt



*Bestimmung einer **DNF**, wobei in b) jeweils " \vee " und " \wedge " vertauscht werden.*

■ Beispiel 2.7

Erzeugen einer KNF aus folgender Formel: $\neg(\neg A \vee B) \vee \neg C$

a) Negationszeichen von außen nach innen ziehen:

Vereinfachung:

≡

.....

b) Alle " \vee "-Zeichen soweit wie möglich nach innen ziehen:

≡

.....

Algorithmus II: Wahrheitstafeln

Gegeben: Wahrheitstafel einer Formel F

Gesucht: Eine zu F äquivalente Formel in DNF bzw. KNF

Bestimmung einer **DNF**:

$a(A_1)$...	$a(A_n)$	$a^*(F)$
0	...	0	.
.
.
1	...	1	.

Vorgehen: Für jede Zeile mit $a^*(F) = 1$ wird eine Konjunktion $(L_1 \wedge \dots \wedge L_n)$ von n Literalen in folgender Weise gebildet:

$$L_i = \begin{cases} A_i, & \text{falls } a(A_i) = 1 \\ \neg A_i, & \text{sonst} \end{cases}$$

Die gesuchte Formel ist die Disjunktion dieser Konjunktionen.

Bestimmung einer **KNF**:

indem die Rollen von falsch und wahr sowie " \vee " und " \wedge ,"
vertauscht werden:

Für jede Zeile mit $a^*(F) = \text{falsch}$ wird eine Disjunktion
($L_1 \vee \dots \vee L_n$) von n Literalen in folgender Weise gebildet:

$$L_i = \begin{cases} A_i, & \text{falls } a(A_i) = 0 \\ \neg A_i, & \text{sonst} \end{cases}$$

Die gesuchte Formel ist die Konjunktion dieser Disjunktionen.



■ Beispiel 2.8

Erzeugen einer KNF / DNF aus der Formel $((A \vee B) \wedge \neg C)$

A	B	C	$((A \vee B) \wedge \neg C)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

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

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



weiterer Ansatz für einen systematischen, häufig auch effizienten Erfüllbarkeitstest:

Resolution:

Voraussetzung: F liegt in KNF vor :

$$F = (L_{1,1} \vee \dots \vee L_{1,n_1}) \wedge \dots \wedge (L_{k,1} \vee \dots \vee L_{k,n_k})$$

$L_{i,j}$ sind Literale

Definition:

Seien $F1 = L_{1,1} \vee \dots \vee L_{1,n_1}$, $F2 = L_{2,1} \vee \dots \vee L_{2,n_2}$ Disjunktionen von Literalen (d.h. Klauseln). Angenommen $L_{1,k} = A$ und $L_{2,j} = \neg A$

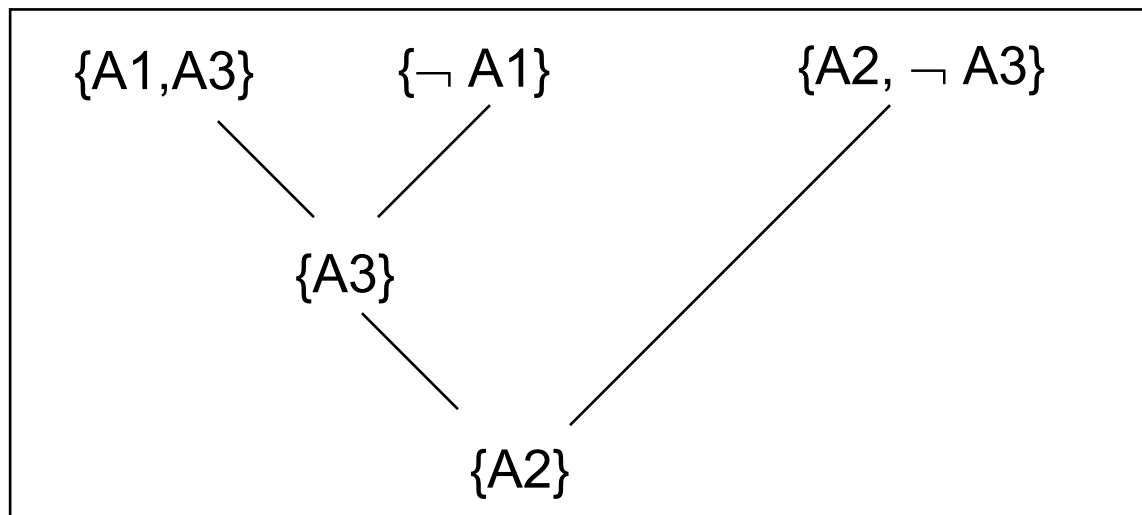
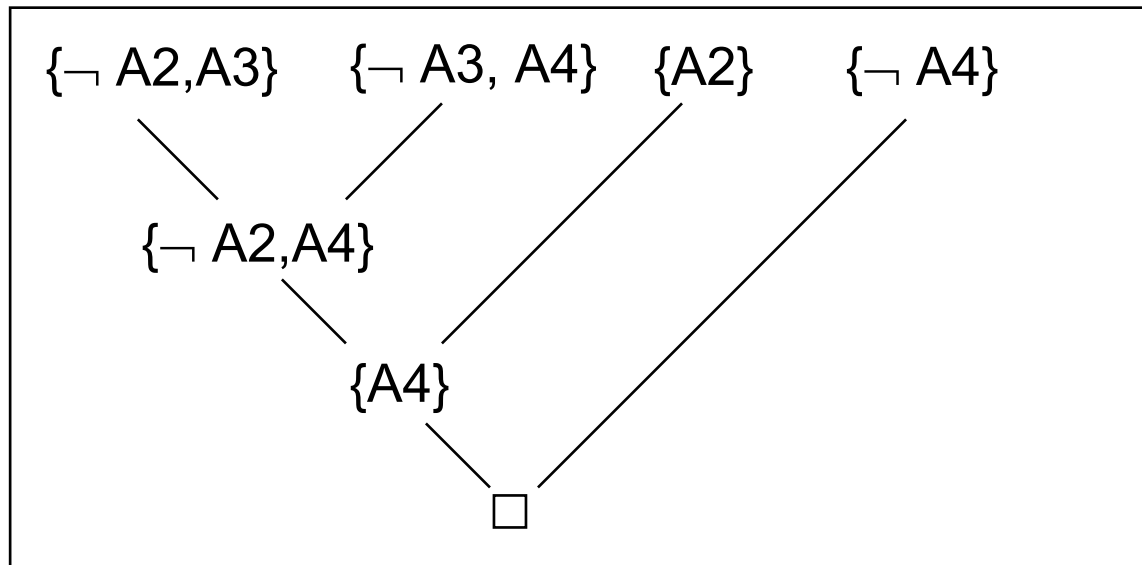
Dann entsteht durch **Resolution** die Klausel

$$L_{1,1} \vee \dots \vee L_{1,k-1} \vee L_{1,k+1} \vee \dots \vee L_{1,n_1} \vee \\ L_{2,1} \vee \dots \vee L_{2,j-1} \vee L_{2,j+1} \vee \dots \vee L_{2,n_2} .$$

Diese Klausel wird **Resolvente** von F1 und F2 genannt.

Falls $F1 = A$ und $F2 = \neg A$, entsteht durch Resolution die leere Klausel \square . Die leere Klausel ist **unerfüllbar** !

■ Beispiel 2.9



Resolutionslemma:

F entstehe durch Resolution aus F1 und F2. Dann gilt:

$$\{F1, F2\} \models F$$

Insbesondere ist $\{F1, F2\}$ unerfüllbar, wenn F unerfüllbar ist.

Resolutionssatz der Aussagenlogik:

Eine Formel in KNF ist genau dann unerfüllbar, wenn durch wiederholte Resolutionen die leere Klausel erzeugt werden kann.



■ Beispiel 2.10



Deduktionstheorem:

$$M \models F \Leftrightarrow M \cup \{\neg F\} \text{ widerspruchsvoll}$$

Regelbasis (3 Axiome) über ein Fahrzeug:

- I) $\text{cabrio} \Leftrightarrow \text{auto} \wedge \text{offen}$
- II) $\text{auto} \Leftrightarrow \text{vierräder}$
- III) $\text{zweiräder} \Leftrightarrow \neg \text{vierräder}$

Modell M

$$C \Leftrightarrow A \wedge O$$

$$A \Leftrightarrow V$$

$$Z \Leftrightarrow \neg V$$

Behauptung:

Ein Cabrio hat nie zwei Räder

Formal:

$$F = (C \Rightarrow \neg Z)$$

Lösung: „Ja“ $\Leftrightarrow M \models F$ *(Modell folgert Formel)*

$$(C \Leftrightarrow A \wedge O) \wedge (A \Leftrightarrow V) \wedge (Z \Leftrightarrow \neg V) \models (C \Rightarrow \neg Z)$$

Umformen in KNF: *(Ja \Leftrightarrow gültig)*

$$\equiv (\neg C \vee A) \wedge (\neg C \vee O) \wedge (\neg A \vee \neg O \vee C) \wedge (\neg A \vee V) \wedge (\neg V \vee A) \wedge (\neg Z \vee \neg V) \wedge (Z \vee V) \models (\neg C \vee \neg Z)$$

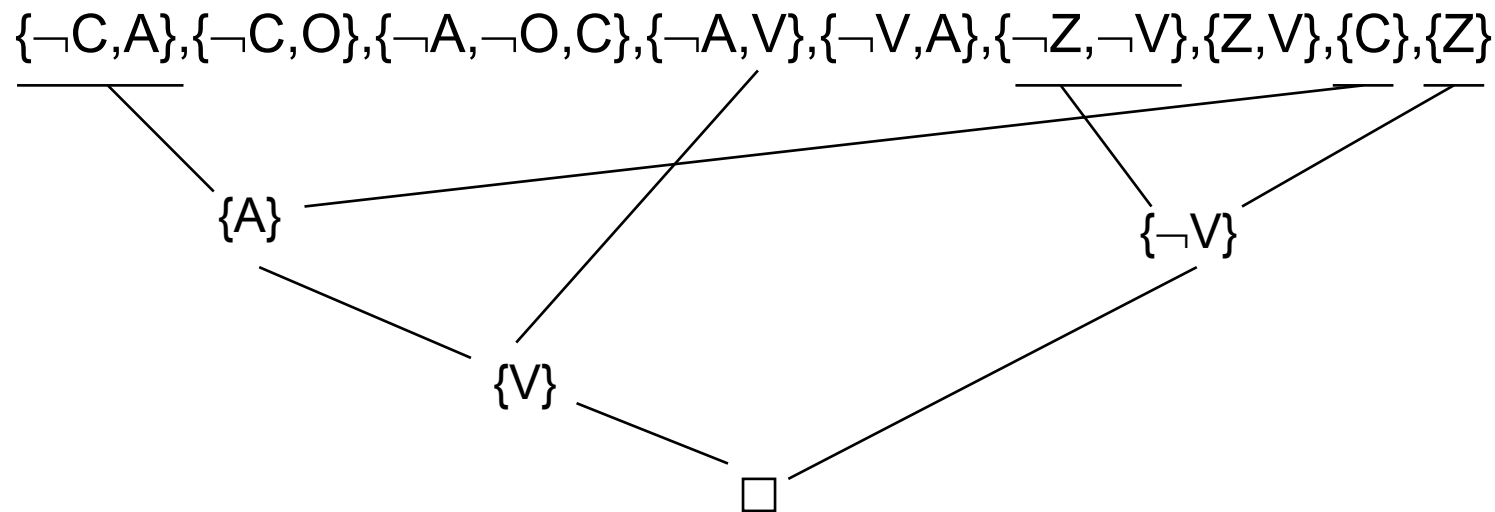
■ Beispiel 2.10

Klauselschreibweise: (Ja $\Leftrightarrow M \cup \neg F$ unerfüllbar)

$$\equiv \{\neg C, A\}, \{\neg C, O\}, \{\neg A, \neg O, C\}, \{\neg A, V\}, \{\neg V, A\}, \{\neg Z, \neg V\}, \{Z, V\} \cup \{\neg(\neg C \vee \neg Z)\}$$

$$\equiv \{\neg C, A\}, \{\neg C, O\}, \{\neg A, \neg O, C\}, \{\neg A, V\}, \{\neg V, A\}, \{\neg Z, \neg V\}, \{Z, V\}, \{C\}, \{Z\}$$

Resolution:



Ergebnis:

- $M \cup \neg F$ ist unerfüllbar
- $\equiv M \models F$ Modell folgert F
- \equiv Ein Cabrio hat nie zwei Räder (es hat immer vier)



1. Informatik: Eine Übersicht

2. Logik (Einführung / Grundlagen)

2.1 Aussagenlogik

2.2 Prädikatenlogik

2.3 Boolesche Algebra

3. Objektorientierte Modellierung

4. Algorithmen und ihre Eigenschaften

5. Entwurfsmethoden von Algorithmen

6. Sortieralgorithmen

7. Dynamische Datenstrukturen



■ Beispiel 2.10

Warum Prädikatenlogik?

Logische Aussagen über und Beziehungen zwischen Objekten.

Aussagenlogik:

Fakten: mietbar, betankt, vermietet

Regeln: $\text{betankt} \wedge \neg \text{vermietet} \Rightarrow \text{mietbar}$

Prädikatenlogik:

Fakten: $\text{betankt}(M1)$, $\neg \text{vermietet}(M1)$, $\neg \text{betankt}(S1)$

Regeln: $\text{betankt}(x) \wedge \neg \text{vermietet}(x) \Rightarrow \text{mietbar}(x)$

Anwendung: $\text{mietbar}(M1)$

→ Prädikatenlogik ist ausdrucksmächtiger

z.B. auch folgende Geschäftslogik möglich:

$\text{vermietet}(M1) \Rightarrow \neg \text{mietbar}(S1)$, $\text{vermietet}(S1) \Rightarrow \neg \text{mietbar}(M1)$

(da der Verleiher immer selbst eines seiner beiden Fahrzeuge nutzt)



AIFB

©AIFB

- Erweiterung der Aussagenlogik
- **Sprache zur Beschreibung der Mathematik und deren philosophischer Grundlage stellt für automatisches Schliessen erforderliche Mittel bereit**
- Untersuchung der "Feinstruktur" von Aussagen, wie etwa:

Für alle $\varepsilon > 0$ gibt es ein n_0 , so dass für alle $n \geq n_0$ gilt: $|f(n) - a| < \varepsilon$

- Man benötigt:
 - **Variablen** (x, y, z, n, \dots)
 - **Funktionszeichen** ($f, g, h, \dots, +, -, *, \dots$)
 - **Prädikatzeichen** ($p, q, r, \dots, <, =, >, \dots$)
 - **Junktoren** ($\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$ (bzw. \equiv))
 - **Quantoren** (\forall, \exists)

Im Folgenden:

- Definition der **Struktur** (bzw. **Syntax**) prädikatenlogischer Formeln
- Definition der **Bedeutung** (bzw. **Semantik**) dieser Formeln



Definition: Syntax der Prädikatenlogik

Gegeben:

- eine Menge **V** von **Variablen**: $V = \{x_1, x_2, \dots\}$,
- eine Menge **FZ** von **Funktionszeichen**
- eine Menge **PZ** von **Prädikatzeichen**
mit: V, FZ, PZ paarweise disjunkt.

- Jedes Funktions-/Prädikatzeichen hat eine **Stelligkeit** $n \geq 0$.
- 0-stellige Funktionszeichen heißen **Konstanten**.
- 0-stellige Prädikatzeichen heißen **Aussagezeichen**.
- $B = (FZ, PZ)$ bezeichnet man als eine **Basis** der Prädikatenlogik.
- Der **Zeichenvorrat Z** über der Basis B ist definiert durch
 $Z = V \cup PZ \cup FZ \cup \{ (,), ., ,, ", \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow, \exists, \forall \}$.
(alle Mengen paarweise disjunkt)



■ Beispiel 2.11

$$x \leq 10 \Rightarrow x+1 \leq 11$$

- x ist Variable
- $+$ ist Funktionszeichen der Stelligkeit 2
- \leq ist Prädikatzeichen der Stelligkeit 2
- die Zahlsymbole 1, 10 und 11 sind Funktionszeichen der Stelligkeit 0, also Konstanten
- „ \Rightarrow “ Implikationspfeil ist logischer Junktor wie in der Aussagenlogik

(1) Die Menge **TB** aller **Terme über der Basis B** ist induktiv definiert durch:

(a) $V \subseteq TB$

Jede Variable ist ein Term.

(b) Wenn f ein Funktionszeichen ($f \in FZ$) der Stelligkeit 0 ist, dann gilt $f \in TB$.

(c) Wenn f ein Funktionszeichen ($f \in FZ$) der Stelligkeit $n > 0$ ist und $t_1, \dots, t_n \in TB$, dann gilt $f(t_1, \dots, t_n) \in TB$.

(d) Nur Zeichenreihen, die wie oben in endlich vielen Schritten erzeugt werden können, sind Terme.



(2) Die Menge **PLB** aller **prädikatenlogischen Formeln über B** ist induktiv definiert durch:

- (a) Wenn P ein Prädikatzeichen der Stelligkeit 0 ist, dann gilt $P \in \text{PLB}$.
- (b) Wenn P ein Prädikatzeichen ($P \in \text{PZ}$) der Stelligkeit $n > 0$ ist und $t_1, \dots, t_n \in \text{TB}$, dann gilt: $P(t_1, \dots, t_n) \in \text{PLB}$.
- (c) Wenn $F, G \in \text{PLB}$, dann sind auch $\neg F, \neg G \in \text{PLB}$, $(F \wedge G) \in \text{PLB}$ und $(F \vee G) \in \text{PLB}$.
- (d) Wenn $F \in \text{PLB}$ und $x \in V$, dann ist auch $\exists x F \in \text{PLB}$ und $\forall x F \in \text{PLB}$.
- (e) Nur Zeichenreihen, die wie oben in endlich vielen Schritten gebildet werden können, sind Formeln.



■ Beispiel 2.12

V	=	$\{x_1, x_2, \dots\}$
FZ	=	$\{f, g, h\}$
PZ	=	$\{P, Q, R\}$

Dabei seien f und P einstellig sowie g, h, Q und R zweistellig.

(1) Beispiele für Terme:

$$x_1$$

$$g(x_1, x_2)$$

$$h(f(x_1), g(x_1, x_2))$$

(2) Beispiele für Formeln:

$$P(x_2)$$

$$\neg P(x_2)$$

$$Q(x_1, x_2)$$

$$P(x_2) \vee \neg Q(x_1, x_2)$$

$$R(h(x_3, x_3), g(x_1, x_2))$$

$$\forall x_2 Q(x_1, x_2) \vee \exists x_3 \neg R(x_1, x_3)$$

Bemerkung:

Eigenschaften von Termen und Formeln können wieder durch Induktion über die Struktur von Termen und Formeln bewiesen werden.



Vereinfachungen:

u, v, w, x, y, z

stehen für Variablen

a, b, c, ...

stehen für Konstanten

f, g, h

stehen für Funktionssymbole

P, Q, R, ...

stehen für Prädikatzeichen



AIFB

©AIFB

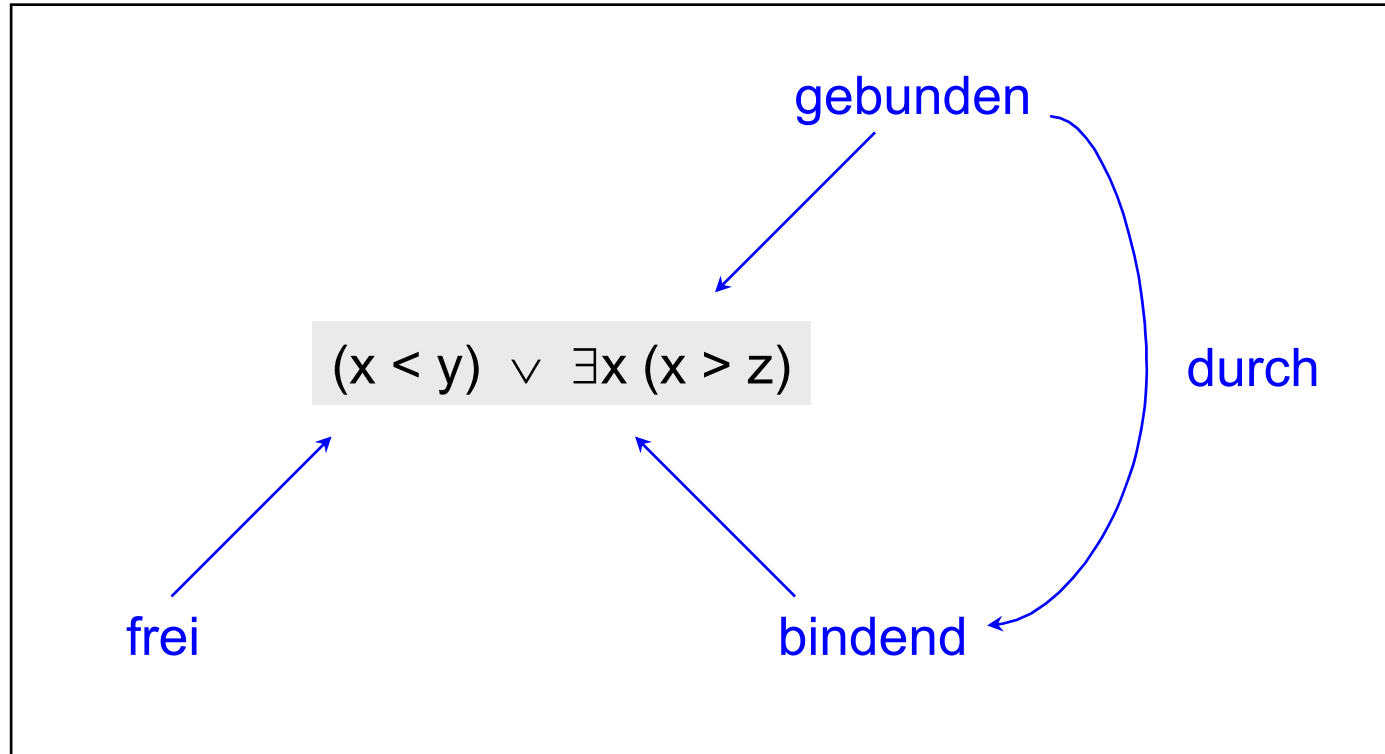
Beobachtung:

Sofern unendlich viele Prädikatzeichen der Stelligkeit 0 verfügbar sind, ist die Aussagenlogik in PLB enthalten.

Bezeichnungen:

- Die Formeln $P(t_1, \dots, t_n)$ aus (2a) und (2b) heißen **atomar**
- Sind F und G Formeln und tritt F als Teil von G auf, so heißt F **Teilformel** von G .
- Ein Vorkommen einer Variablen x in einer Formel F heißt
 - **bindend**, wenn x in F direkt hinter einem \exists oder \forall vorkommt,
 - **gebunden**, wenn F eine Teilformel der Form $\exists x G$ oder $\forall x G$ hat und x in G vorkommt,
 - **frei**, sonst.
- Eine Formel, in der keine Variable frei vorkommt, heißt **geschlossen** oder eine **Aussage**.

■ Beispiel 2.13



F sei: $(\exists x Q(x, f(y)) \vee \neg \forall y R(y, g(z, f(z))))$

alle Teilformeln von F:

$$\begin{aligned} & (\exists x Q(x, f(y)) \vee \neg \forall y R(y, g(z, f(z)))) \\ & \exists x Q(x, f(y)) \\ & Q(x, f(y)) \\ & \neg \forall y R(y, g(z, f(z))) \\ & \forall y R(y, g(z, f(z))) \\ & R(y, g(z, f(z))) \end{aligned}$$

alle Terme, die in F vorkommen:

x	gebunden	}	F ist also keine Aussage
y	einmal frei, einmal gebunden		
z	zweimal frei		
f(y)			
f(z)			
g(z, f(z))			



Definition der Semantik prädikatenlogischer Formeln durch:

- Festlegung eines Wertebereichs für Variablen,
- Zuordnung von Funktionen über dem Wertebereich zu Funktionszeichen,
- Zuordnung von Relationen über dem Wertebereich zu Prädikatzeichen.



Definition: Semantik der Prädikatenlogik

Geg.: Basis B

Eine **Struktur** für PLB ist ein Paar $\mathbf{s} = (\mathbf{U}, I)$ mit

- \mathbf{U} ist eine nichtleere Menge
(**Werte-** oder **Individuenbereich** oder **Universum** von \mathbf{s}).
- I ist eine Abbildung (genannt: **Interpretation** von \mathbf{s}), die
 - jeder Variablen x_i einen Wert $I(x_i) \in \mathbf{U}$
 - jedem Funktionszeichen f der Stelligkeit n eine Funktion $I(f): \mathbf{U}^n \rightarrow \mathbf{U}$,
 - jedem Prädikatzeichen P der Stelligkeit n eine Relation $I(P) \subseteq \mathbf{U}^n$ zuordnet.



D.h.: Definitionsbereich von I ist Teilmenge von $V \cup FZ \cup PZ$.



■ Beispiel 2.14

$$F = \forall x \ P(x, f(x)) \wedge Q(g(a, b))$$

Eine mögliche Struktur s könnte die folgende sein:

$$\mathbf{U} = \mathbf{IN}$$

$$I(a) = 2$$

$$I(b) = 3$$

$$I(f) = \text{die Nachfolgerfunktion auf } \mathbf{U}, \text{ also } I(f)(n) = n+1$$

$$I(g) = \text{die Additionsfunktion auf } \mathbf{U}, \text{ also } I(g)(m,n) = m+n$$

$$I(P) = \{ (m, n) \mid m, n \in \mathbf{U} \text{ und } m < n \}$$

$$I(Q) = \{ n \in \mathbf{U} \mid n \text{ ist Primzahl} \}$$

Bedeutung von F unter s :

Für jede natürliche Zahl x gilt :
 $x < x+1$ und $2+3$ ist Primzahl.

Schreibweise:

Sei s eine Struktur, $x \in V$, $d \in \mathbf{U}$.

Dann bezeichnet $s[x/d]$ diejenige Struktur, die sich von s nur in der Interpretation der Variablen x unterscheidet, und zwar soll gelten:

$$I[x/d](x) = d$$

d.h. der Variablen x wird der Wert d zugewiesen (gleichgültig, welchen Wert $I(x)$ hat).



Definition: Semantik von Termen und Formeln

Für jede Struktur s über der Basis $B = (FZ, PZ)$ sei

$$\begin{array}{l} s^* : TB \rightarrow \mathbf{U} \text{ und} \\ s^* : PLB \rightarrow \{0, 1\} \end{array}$$

durch Induktion über die Struktur von Termen und Formeln definiert:

(1) Für Terme:

$$(a) \quad s^*(x_i) = I(x_i)$$

$$(b) \quad s^*(f(t_1, \dots, t_n)) = I(f)(s^*(t_1), \dots, s^*(t_n))$$



- Für Formeln

$$(a) \quad s^*(P(t_1, \dots, t_n)) = \begin{cases} 1, & \text{falls } (s^*(t_1), \dots, s^*(t_n)) \in I(P) \\ 0, & \text{sonst} \end{cases}$$

$$\begin{array}{l} (b) \quad s^*(\neg F) \\ (c) \quad s^*((F \wedge G)) \\ (d) \quad s^*((F \vee G)) \end{array} \left. \vphantom{\begin{array}{l} (b) \\ (c) \\ (d) \end{array}} \right\} \text{wie in der Aussagenlogik}$$

$$(e) \quad s^*(\forall x F) = \begin{cases} 1, & \text{falls für jedes } d \in \mathbf{U}: \\ & s[x/d]^*(F) = 1 \\ 0, & \text{sonst} \end{cases}$$

$$(f) \quad s^*(\exists x F) = \begin{cases} 1, & \text{falls es ein } d \in \mathbf{U} \text{ gibt, mit:} \\ & s[x/d]^*(F) = 1 \\ 0, & \text{sonst} \end{cases}$$



■ Beispiel 2.15

Sei $\mathbf{U} = \mathbb{N}_0$ und $I(<)$, $I(+)$, ... wie üblich definiert.
Außerdem $I(x) = 1$, $I(y) = 2$

Dann gilt:

$$\begin{aligned} s^*((x+1) < y) &= 0, \text{ denn:} \\ s^*((x+1) < y) &= I(<)(s^*(x+1), s^*(y)) \\ &= I(<)(I(+)(I(x), 1), I(y)) \\ &= I(<)(2, 2) = 0 \end{aligned}$$

$$\text{aber: } s^*(\exists x (x+1) < y) = 1$$

$$\text{denn für } d=0 \text{ gilt: } s[x/0]^*((x+1) < y) = 1$$

$$\text{folglich existiert ein } d \text{ mit } s[x/d]^*((x+1) < y) = 1$$

$$\text{deshalb ist } s^*(\exists x (x+1) < y) = 1$$

Beachte:

$s^*(\exists x (x+1) < y)$ ist offensichtlich unabhängig von $I(x)$

Intuitive Erklärung:

$s^*(F)$ hängt nur von den Werten $I(x)$ derjenigen Variablen x ab, die **frei** in F vorkommen.





Satz: **Koinzidenzsatz der Prädikatenlogik**

Seien:

- F eine Formel;
- s_1, s_2 zwei Strukturen, für die gilt:

$$U_1 = U_2$$

I_1 und I_2 stimmen überein

- auf allen Funktionszeichen von F
- auf allen Prädikatszeichen von F
- auf allen Variablen, die in F frei vorkommen

Dann gilt:

$$s_1^*(F) = s_2^*(F)$$



Beweis: Durch Induktion über Struktur der Terme / Formeln

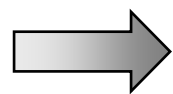
Definitionen / Bezeichnungen

Übernahme vieler Definitionen und Schreibweisen aus der Aussagenlogik (d.h. ersetze die Belegung a durch die Struktur s):
($F \in \text{PLB}$)

- F ist **gültig** unter der Struktur s : $s \models F$ (s ist **Modell** von F)
- F ist **erfüllbar**: $\exists s$ mit $s \models F$
- F ist **unerfüllbar**: $\neg \exists s$ mit $s \models F$
- F ist **allgemeingültig**: $\models F$
- s ist **Modell einer Formelmenge** M ; $s \models M$
- M ist **erfüllbar**: $\exists s$ mit $s \models M$
- M ist **widerspruchsvoll**: $\neg \exists s$ mit $s \models M$
- F ist logische **Folgerung** von M : $M \models F$
- F und G sind **äquivalent**: $F \Leftrightarrow G$, bzw. $F \equiv G$
(d.h. für jede Struktur s gilt: $s^*(F) = s^*(G)$)

Satz: ÄquivalenzenFür beliebige Formeln F, G gilt:

(1)	$\neg \forall x F$	\equiv	$\exists x \neg F$	
	$\neg \exists x F$	\equiv	$\forall x \neg F$	
(2)	$\forall x (F \wedge G)$	\equiv	$(\forall x F \wedge \forall x G)$	
	$\exists x (F \vee G)$	\equiv	$(\exists x F \vee \exists x G)$	
Aber:	$\forall x (F \vee G)$	$\not\equiv$	$(\forall x F \vee \forall x G)$	
	$\exists x (F \wedge G)$	$\not\equiv$	$(\exists x F \wedge \exists x G)$	
(3)	$(\forall x F \wedge G)$	\equiv	$\forall x (F \wedge G)$	falls x nicht frei in G vorkommt
	$(\forall x F \vee G)$	\equiv	$\forall x (F \vee G)$	
	$(\exists x F \wedge G)$	\equiv	$\exists x (F \wedge G)$	



Die Äquivalenzen (1) und (3) können (wenn die Voraussetzung für (3) erfüllt ist) benutzt werden, um Quantoren "nach außen zu ziehen".

Satz: Äquivalenzen (Fortsetzung)

(4)	$\forall x \forall y F$	\equiv	$\forall y \forall x F$
	$\exists x \exists y F$	\equiv	$\exists y \exists x F$
Aber:	$\forall x \exists y F$	$\not\equiv$	$\exists y \forall x F$



■ Beispiel 2.16

$x \in \mathbb{N}$

F sei: x ist geradzahlig,

G sei: x ist ungeradzahlig.

$s^*(\forall x (F \vee G)) = \dots\dots\dots$ (da jedes x gerade oder ungerade ist)

aber:

$s^*(\forall x F \vee \forall x G) = \dots\dots\dots$ (da weder alle x gerade noch alle x ungerade sind)

x, F, G wie oben:

$s^*(\exists x (F \wedge G)) = \dots\dots\dots$ (da kein x gerade und ungerade ist)

aber:

$s^*(\exists x F \wedge \exists x G) = \dots\dots\dots$ (da (mindestens) ein x gerade und (mindestens) ein x ungerade ist)



Bemerkung:

In der Prädikatenlogik können im Gegensatz zur Aussagenlogik Eigenschaften von Formeln **nicht** mit (endlichen) Wahrheitstafeln überprüft werden.

Frage:

Gibt es Verfahren, um die Eigenschaften von Formeln zu überprüfen?



Satz: Unentscheidbarkeit der Prädikatenlogik

Es gibt keinen Algorithmus, der angesetzt auf eine beliebige Formel

- ja ausgibt, genau dann wenn die Formel allgemeingültig ist,
- nein ausgibt, sonst.

Man sagt: Die Menge aller allgemeingültigen Formeln der Prädikatenlogik ist **unentscheidbar**.

Folgerung:

Auch die Menge der erfüllbaren Formeln ist unentscheidbar, denn:

$$F \text{ allgemeingültig} \Leftrightarrow \neg F \text{ unerfüllbar}$$

Satz: Semi-Entscheidbarkeit der Prädikatenlogik

Es gibt einen Algorithmus, der angesetzt auf eine beliebige prädikatenlogische Formel

- mit "ja" terminiert, wenn die Formel allgemeingültig ist;
- nicht terminiert, sonst.

Weitere Umformungen prädikatenlogischer Formeln:

- Umbenennung gebundener Variablen
- Substitution einer Variablen durch eine andere Variable
- Substitution einer Variablen durch einen Term
- Normalformen

Intuition: Der Name einer gebundenen Variablen ist variierbar.

■ Beispiel 2.17

Sei $F = \forall x \exists y x < y$:

- Umbenennung von x in u führt zu der äquivalenten Formel

$F' =$
.....

- aber: Umbenennung von x in y führt zu einer nicht äquivalenten Formel

$H =$
.....

Satz: Umbenennung gebundener Variablen

Sei $F = \forall x G$ (bzw. $F = \exists x G$) und sei y eine Variable, die nicht in G vorkommt.

Dann ist F äquivalent zu $\forall y G[x/y]$ (bzw. zu $\exists y G[x/y]$),

wobei $G[x/y]$ die Formel ist, die aus G entsteht, indem jedes freie Vorkommen von x in G durch y ersetzt wird.

■ Beispiel 2.18

In Beispiel 3.16 sei $G = \exists y x < y$, u kommt nicht in G vor.
Es ist $G[x/u] = \exists y u < y$, also ist F äquivalent zu $\forall u \exists y u < y$.

Auf y - anstelle von u - ist der Satz nicht anwendbar, da y in G vorkommt.

Substitution

- Eine Variable steht für einen beliebigen Wert.
Folglich müsste es sinnvoll möglich sein, einen Term für eine Variable einzusetzen.
- Insbesondere:
Wenn $\forall x F$ (in einer bestimmten Struktur) gilt, dann sollte auch $F[x/t]$ für jeden beliebigen Term t gelten, wobei $F[x/t]$ aus F entsteht, indem man x durch t ersetzt.



■ Beispiel 2.19

- Wenn $\forall x \exists y x < y$ in einer Struktur s gilt, dann gilt in s auch:
 $\exists y x+1 < y$ (Substitution von x durch $x+1$)
- **aber:** Durch Substitution von x durch $y+1$ ergibt sich $\exists y y+1 < y$
(falsch!!)
- Problem:
Die Variable y im Term $y+1$ wird durch die Bindung $\exists y \dots$
verknüpft. Dieser "Bindungskonflikt" muss aufgelöst werden.



Vorgehen: Substitution einer Variablen durch einen Term

Sei F eine Formel, x eine Variable, t ein Term.

Man erhält die Formel $F[x/t]$ wie folgt:

- Durch Umbenennung gebundener Variablen erzeugt man zunächst eine zu F äquivalente Formel F' , für die gilt:
Keine der in t vorkommenden Variablen kommt in F' bindend vor.
- Dann ersetzt man jedes freie Vorkommen von x in F' durch t .

■ Beispiel 2.20

Sei $F = \exists y \ x < y$ und $t = y + 1$

Dann ist $F' =$

.....

also $F[x/t] =$

.....

Satz:

Für jede Formel F , jede Variable x und jeden Term t gilt:

$$\begin{array}{l} \forall x F \quad \Rightarrow \quad F[x/t] \\ F[x/t] \quad \Rightarrow \quad \exists x F \end{array}$$



Normalformen:

Lemma:

Zu jeder Formel F gibt es eine äquivalente Formel G in bereinigter Form.

Hierbei heißt eine Formel **bereinigt**, wenn es keine Variable gibt, die in der Formel sowohl gebunden als auch frei vorkommt, und wenn hinter allen vorkommenden Quantoren verschiedene Variablen stehen.

■ Beispiel 2.21

$$F = \forall x \exists y P(x, f(y)) \wedge \forall y (Q(x, y) \vee R(x))$$

$$G = \forall u \exists v P(u, f(v)) \wedge \forall y (Q(x, y) \vee R(x))$$

G ist bereinigte Formel für F



Definition: Pränex-Normal

Eine Formel F ist in **Pränex-Normalform** (PNF), wenn
 $F = Q_1 x_1 \dots Q_n x_n G$ ($n > 0$),

wobei die Q_i Quantoren sind und in G kein Quantor mehr vorkommt.

Liegt eine Formel in PNF vor und ist sie gleichzeitig bereinigt, so spricht man von einer bereinigten PNF.

Satz:

Zu jeder Formel F existiert eine äquivalente (und bereinigte) Formel in Pränex-Normalform.



1. Informatik: Eine Übersicht

2. Logik (Einführung / Grundlagen)

2.1 Aussagenlogik

2.2 Prädikatenlogik

2.3 **Boolesche Algebra**

3. Objektorientierte Modellierung

4. Algorithmen und ihre Eigenschaften

5. Entwurfsmethoden von Algorithmen

6. Sortieralgorithmen

7. Dynamische Datenstrukturen



Ursprung: George **Boole** (1815-1864), Mathematiker

- Math. Formalisierung der Gesetze der Logik
- Kernstück moderner Informatik
- Mächtigkeit & Einfachheit

2.3.1 Definition einer Booleschen Algebra

Eine **Boolesche Algebra** ist eine algebraische Struktur $(M; \cdot, +, ')$ (*algebraische Struktur = Menge mit Verknüpfung(en)*) bestehend aus

- einer Menge M mit mindestens zwei Elementen ($|M| \geq 2$)
- den zweistelligen Verknüpfungen
 - $\cdot : M \times M \rightarrow M, (a,b) \rightarrow a \cdot b$ (**Boolesches Produkt**)
 - $+: M \times M \rightarrow M, (a,b) \rightarrow a + b$ (**Boolesche Summe**)
- und der einstelligen Verknüpfung
 - $' : M \rightarrow M, a \rightarrow a'$ (**Boolesches Komplement**)



Die Verknüpfungen genügen den folgenden Axiomen:

(BA1): "." und "+" sind kommutativ :

$$(K\cdot): \quad \forall a, b \in M: \quad a \cdot b = b \cdot a$$

$$(K+): \quad \forall a, b \in M: \quad a + b = b + a$$

} Kommutativgesetze

(BA2): "." und "+" sind wechselseitig distributiv:

$$(D\cdot): \quad \forall a, b, c \in M: a \cdot (b + c) = a \cdot b + a \cdot c$$

$$(D+): \quad \forall a, b, c \in M: a + (b \cdot c) = (a + b) \cdot (a + c)$$

} Distributivgesetze



(BA3): es gibt neutrale Elemente 0 und 1:

$$\exists 1 \in M \quad \forall a \in M: \quad a \cdot 1 = a$$

$$\exists 0 \in M \quad \forall a \in M: \quad a + 0 = a$$

(BA4): a' ist komplementär zu a in der folgenden Weise:

$$(C\cdot): \quad \forall a \in M: \quad a \cdot a' = 0$$

$$(C+): \quad \forall a \in M: \quad a + a' = 1$$

} Komplementgesetze

Bemerkung:

Erste, vollständige, formale Festlegung einer Booleschen Algebra durch Huntington (1904).

(BA1) bis (BA4): Huntingtonsche Axiome.



Satz: (Dualitätsprinzip)

Zu jeder aus (BA1) - (BA4) herleitbaren Formel existiert eine "duale" Formel, die ebenfalls gilt. Sie entsteht durch Vertauschung von "." mit "+" und von 1 mit 0.

Beweis:

Die einzelnen Aussagen der Axiome (BA1) - (BA4) sind dual zueinander. Daher gibt es zu jeder Herleitung einer Formel F auch die duale Herleitung, deren Ergebnis die zu F duale Formel ist.

Für die Herleitung weiterer Sätze bedeutet das:

Mit dem Beweis einer Formel ist auch die dazu duale Formel bewiesen.

Satz:

$$\forall a \in M : a \cdot 0 = 0$$

$$\forall a \in M : a + 1 = 1$$

Beweis:

Es genügt zu zeigen: $a+1 = 1$.

Nach dem Dualitätsprinzip ist dann auch $a \cdot 0 = 0$ bewiesen.

$$\begin{aligned}
 1 &= \dots\dots\dots \\
 &= \dots\dots\dots \\
 &= \dots\dots\dots \\
 &= \dots\dots\dots \\
 &= \dots\dots\dots
 \end{aligned}$$

Bemerkung:

Die Gesetze von Axiom (BA3) und dem obigen Satz bezeichnet man auch als 0-1-Gesetze.

(N ·): $\forall a \in M: a \cdot 1 = a, \quad a \cdot 0 = 0$

(N+): $\forall a \in M: a + 1 = 1, \quad a + 0 = a$





Weitere Gesetze

Idempotenzgesetze:

$$(I\cdot): \forall a \in M : a \cdot a = a$$

$$(I+): \forall a \in M : a + a = a$$

wenn etwas als wahr bekannt ist, kann dieses Wissen nicht durch Wiederholung erweitert werden.

Assoziativgesetze:

$$(A\cdot): \forall a, b, c \in M : a \cdot (b \cdot c) = (a \cdot b) \cdot c$$

$$(A+): \forall a, b, c \in M : a + (b + c) = (a + b) + c$$

De Morgansche Regeln:

$$(M\cdot): \forall a, b \in M : (a \cdot b)' = a' + b'$$

$$(M+): \forall a, b \in M : (a + b)' = a' \cdot b'$$

Absorptionsgesetze:

$$(Ab \cdot): \quad \forall a, b \in M : a \cdot (a + b) = a$$

$$(Ab+): \quad \forall a, b \in M : a + (a \cdot b) = a$$

Doppeltes Boolesches Komplement:

$$(KK): \quad \forall a \in M : a'' = a$$

Komplementarität der neutralen Elemente:

$$(K0): \quad 0' = 1$$

$$(K1): \quad 1' = 0$$

(Folgerung: Eindeutigkeit der neutralen Elemente)



Beispiel: (Eine Mengenalgebra als Boolesche Algebra)

Sei eine Menge $M = \{a, b, c, \dots\}$ gegeben und sei ferner

$$\begin{aligned} P(M) &:= \{ m \mid m \subseteq M \} \text{ ("Potenzmenge" von } M) \\ &= \{ \emptyset, \{a\}, \{b\}, \dots, \{a, b\}, \{a, c\}, \dots, \{a, b, c\}, \dots, M \} \end{aligned}$$

Wir ersetzen die Verknüpfungen

- durch \cap (*Mengendurchschnitt*)
- + durch \cup (*Mengenvereinigung*) und
- ' durch $-$ (*Komplementärmenge; $\bar{m} := M \setminus m, m \in P(M)$*)



Dann gilt für beliebige Elemente $m_1, m_2, m_3 \in P(M)$:

- $m_1 \cap m_2 = m_2 \cap m_1$ und $m_1 \cup m_2 = m_2 \cup m_1$ (BA1)
- $m_1 \cap (m_2 \cup m_3) = (m_1 \cap m_2) \cup (m_1 \cap m_3)$ (BA2)
 $m_1 \cup (m_2 \cap m_3) = (m_1 \cup m_2) \cap (m_1 \cup m_3)$
- M ist neutrales Element für \cap (BA3)
 \emptyset ist neutrales Element für \cup
- $m_1 \cap \overline{m_1} = \emptyset$ und $m_1 \cup \overline{m_1} = M$ (BA4)

$\Rightarrow (P(M); \cap, \cup, \neg)$ ist eine Boolesche Algebra.



Standardbeispiel: Aussagenlogik

Sei $M = \{\text{falsch, wahr}\}$ bzw. $M = \{0, 1\}$

Wir ersetzen die Verknüpfungen

- durch \wedge (Konjunktion)
- + durch \vee (Disjunktion)
- ' durch \neg (Negation; $\neg\text{wahr} = \text{falsch}$ sowie $\neg\text{falsch} = \text{wahr}$)



Dann gilt für beliebige Elemente $m_1, m_2, m_3 \in (M)$:

$$(BA1): m_1 \wedge m_2 = m_2 \wedge m_1$$

$$m_1 \vee m_2 = m_2 \vee m_1$$

$$(BA2): m_1 \wedge (m_2 \vee m_3) = (m_1 \wedge m_2) \vee (m_1 \wedge m_3)$$

$$m_1 \vee (m_2 \wedge m_3) = (m_1 \vee m_2) \wedge (m_1 \vee m_3)$$

(BA3): wahr ist neutrales Element für \wedge

falsch ist neutrales Element für \vee

$$(BA4): \text{wahr} \wedge \neg \text{wahr} = \text{falsch}$$

$$\text{wahr} \vee \neg \text{wahr} = \text{wahr}$$

$\Rightarrow (M; \wedge, \vee, \neg)$ ist eine Boolesche Algebra



3.3.2 Normalformen

3.3.2.1 Boolesche Funktionen und Boolesche Terme

Im folgenden:

$M = IB = \{0,1\}$ (zweielementige Boolesche Algebra)

0 und 1 nennt man **Boolesche Konstanten**

Verwendung von **Booleschen Variablen** mit Wertebereich IB

Zuordnung konkreter Werte an die Booleschen Variablen a, b, c, \dots heißt **Belegung** der Booleschen Variablen a, b, c, \dots



Des weiteren vereinbaren wir **Vorrangregeln**:
(zur Abkürzung der Schreibweise)

() vor ' , vor · , vor +

Bei Hintereinanderausführung gleicher Verknüpfungen wird von links nach rechts abgearbeitet

· kann weggelassen werden, wenn keine Verwechslungen mit anderen Variablen möglich sind, d.h. $a b = a \cdot b$

Beispiel:

$$a + (b \cdot c) = a + b \cdot c = a + b c$$

$$(a \cdot b') + (c \cdot d)' = a b' + (c d)'$$

$$(a + b) \cdot (a + c) = (a + b) (a + c)$$



Definition: (Boolesche Funktion)

Eine Abbildung $f : \mathbb{B}^n \rightarrow \mathbb{B}$ heißt Boolesche Funktion.

Eine Boolesche Funktion kann durch eine 2^n -zeilige Tabelle, die Wertetabelle, vollständig beschrieben werden.
(entspricht der *Wahrheitstafel* in der Aussagenlogik)

Beispiel ($n = 3$):

Seien a_1, a_2, a_3 Boolesche Variablen

a_1	a_2	a_3	$f(a_1, a_2, a_3)$	linke Seite einer Wertetabelle: alle 2^n möglichen verschiedenen Belegungen der Booleschen Variablen a_1, a_2, \dots, a_n
0	0	0	-	
0	0	1	-	
0	1	0	-	rechte Seite einer Wertetabelle: die zugehörigen Funktionswerte
0	1	1	-	
1	0	0	-	(2^n Einzelwerte, für jeden "0" oder "1" möglich $\Rightarrow 2^{2^n}$ verschiedene Funktionen f)
1	0	1	-	
1	1	0	-	
1	1	1	-	



Definition: (Boolescher Term, Boolescher Ausdruck)

Sei $V = \{a_1, a_2, \dots, a_n\}$ eine Menge Boolescher Variablen,
 $S = \{0, 1, \cdot, +, ', (,)\}$ eine Menge von Symbolen und
 $V \cap S = \emptyset$.

- (1) Jede Konstante 0 und 1 und jede Variable a_i ist ein Boolescher Term.
- (2) Sind A und B Boolesche Terme, so auch A' , $(A \cdot B)$ und $(A + B)$.
- (3) Nur Zeichenreihen, die sich mit (1) und (2) in endlich vielen Schritten konstruieren lassen, sowie deren "Abkürzungen" gemäß den Vorrangregeln sind Boolesche Terme.

(vergleiche: Definition der *Formeln der Aussagenlogik*)



Beispiel: Boolescher Term

Seien a, b, c Boolesche Variablen

.....

.....

.....

.....

.....



AIFB

©AIFB

Bemerkung:

Ein Boolescher Term mit n verschiedenen Variablen realisiert genau eine n -stellige Boolesche Funktion.

Zwei Boolesche Terme sind nur dann **gleich**, wenn sie zeichenweise übereinstimmen, sonst **verschieden**.
("syntaktische Gleichheit")

Beispiel: (verschiedene Boolesche Terme)

$(a + b), (b + a) \quad / \quad 0, 1 \cdot 0 \quad / \quad a \cdot a, a$



Definition: (Äquivalenz Boolescher Terme)

Zwei Boolesche Terme A, B sind **äquivalent**,
geschrieben $A = B$, wenn sie bei gleicher Belegung gemeinsamer
Variablen stets das gleiche Resultat haben. ("semantische
Gleichheit")

Beispiel: (äquivalente Boolesche Terme)

$$(a' + c)' + a' b' = a' b' + a c = (a + b)' + (a \cdot c)$$

Offensichtlich sind zwei Boolesche Terme insbesondere dann
äquivalent, wenn sie (syntaktisch) gleich sind.

Für **eine** Boolesche Funktion gibt es viele verschiedene,
äquivalente Boolesche Terme.

gesucht: Standarddarstellung von Booleschen Funktionen
durch Boolesche Terme (für Verwendung von Algorithmen),
sogenannte **Normalformen**.



3.3.2.2 Disjunktive und konjunktive Normalformen

Definitionen und Bezeichnungen:

(1) Seien a_1, a_2, \dots, a_n paarweise verschiedene Boolesche Variablen und $x_i = a_i$ oder $x_i = a_i'$ ($i = 1, \dots, n$).

Dann nennt man

das Boolesche Produkt $x_1 \cdot x_2 \cdot \dots \cdot x_n$

auch n-stelligen **Konjunktionsterm**,

die Boolesche Summe $x_1 + x_2 + \dots + x_n$

auch n-stelligen **Disjunktionsterm**.

(2) 0-stelliger Konjunktionsterm: Boolesche Konstante 1
0-stelliger Disjunktionsterm: Boolesche Konstante 0



(3) Seien K_1, K_2, \dots, K_m paarweise verschiedene Konjunktionsterme und D_1, D_2, \dots, D_m paarweise verschiedene Disjunktionsterme.

Dann heißt der Boolesche Term

$K_1 + K_2 + \dots + K_m$ **disjunktive Normalform** (DN)

$D_1 \cdot D_2 \cdot \dots \cdot D_m$ **konjunktive Normalform** (KN).

(Die K_i, D_j können verschieden-stellig sein.)

(Sprechweise auch: Ein Boolescher Term
"ist in ...Normalform.")



Beispiel: $n = 3$; a, b, c Boolesche Variablen

(DN):

- $a b c' + a c + a' c$
- $a' b c + a b' c + a b c' + a' b' c + a b' c' + a' b c'$

(KN):

- $(a + b + c') (a + c) (a + b' + c)$
- $(a' + b' + c') (a' + b + c') (a' + b' + c) (a + b + c')$

weder DN noch KN: $(a c') (b + c)$



Satz:

Zu jedem Booleschen Term A gibt es eine zu A äquivalente DN (KN).

Diese ist i. a. nicht eindeutig.

Beweis:

"i.a. nicht eindeutig":

$$\begin{aligned} \text{z.B.: } f : IB^4 &\rightarrow IB \\ (a, b, c, d) &\rightarrow b(c' + ad) \\ &= b c' + a b d = b c' + a b d = a' b c' + a b c' + a b d = \dots \\ &\text{(alles DN)} \end{aligned}$$

für KN analog

"es gibt eine:" durch Angabe eines Algorithmus.



Algorithmus zur Herstellung einer DN (KN) eines Terms:

- (1) Anwendung der De Morganschen Regeln, bis Negation nur noch bei Variablen
- $$(a+b)' = a' \cdot b'$$
- $$(a \cdot b)' = a' + b'$$
- (2) Anwendung des Distributivgesetzes (D \cdot) ((D+))
 des Komplementgesetzes (C \cdot) ((C+))
 des 0-1-Gesetzes (N \cdot) ((N+)) und
 des Idempotenzgesetzes (I \cdot) ((I+)) .
- $$a \cdot (b+c) = a \cdot b + a \cdot c$$
- $$a \cdot a' = 0$$
- $$a \cdot 1 = a, a \cdot 0 = 0$$
- $$a \cdot a = a$$
- (3) Zusammenfassung mehrfach auftretender Konjunktionsterme (Disjunktionsterme), sowie Weglassen von 0 (1).

(vgl. Algorithmen zur Bestimmung von Normalformen logischer Formeln)

