Logik

Studiengang „Informatik“ und „Technoinformatik“ SS'02

Prof. Dr. Madlener
Universität Kaiserslautern

Vorlesung:
Mi 11.45-13.15 52/207

- Informationen
  www-madlener.informatik.uni-kl.de/ag-madlener/teaching/ss2002/logik
- Grundlage der Vorlesung: Skript
  Einführung in die Logik und Korrektheit von Programmen.
- Bewertungsverfahren:
  Aufsichtsarbeit: max. 30 Punkte,
  Abschlussklausur: max. 70 Punkte.
- Aufsichtsarbeit: (5.06 8 - 11 Mensa)
- Erste Abschlussklausur (19.07 16-19 Mensa)
- Übungen: Hörsaalübungen Di. 10.00-11.30 (Vorlesung 30.4)

Inhaltsverzeichnis

1 Einleitung: Einführung in die Logik

Methoden zur Lösung von Problemen mit Hilfe von Rechnern Formalisierung ($\equiv$ Festlegung)

Logik:: „Lehre vom folgenrichtigen Schließen“ bzw. „Lehre von formalen Beziehungen zwischen Denkinhalten“

Zentrale Fragen: Wahrheit und Beweisbarkeit von Aussagen $\rightsquigarrow$ **Mathematische Logik**.

Logik in der Informatik:

Aussagenlogik: Boolsche Algebra. Logische Schaltkreise (Kontrollsystemen), Schaltungen, Optimierung.

Prädikatenlogik: Spezifikation und Verifikation von Softwaresystemen.

1. Semantik von Programmiersprachen (Hoarscher Kalkül).
2. Spezifikation von funktionalen Eigenschaften.
3. Verifikationsprozess bei der SW-Entwicklung. Beweise von Programmeigenschaften.
4. Spezielle Programmiersprachen (z.B. PROLOG)

Automatisierung des logischen Schließens

1. Automatisches Beweisen (Verfahren,...)
2. Grundlagen von Informationsystemen (Verarbeitung von Wissen, Reasoning,...)

Voraussetzungen

1. Mathematische Grundlagen. Mengen, Relationen, Funktionen. Übliche Formalisierungen: „Mathematische Beweise“, Mathematische Sprache, d.h. Gebrauch und Bedeutung der üblichen Operatoren der naiven Logik. Also Bedeutung von **nicht, und, oder, impliziert, äquivalent, es gibt, für alle**.
2. Grundlagen zur Beschreibung formaler Sprachen. Grammatiken oder allgemeiner Kalküle (Objektmenge und Regeln zur Erzeugung neuer Objekte aus bereits konstruierter Objekte), Erzeugung von Mengen, Relationen und Funktionen, Hüllenoperatoren (Abschluss von Mengen bzgl. Relationen).
3. Vorstellung von Berechenbarkeit, d.h. entscheidbare und rek.aufzählbare Mengen, Existenz nicht entscheidbarer Mengen und nicht berechenbarer Funktionen.

Berechnungsmodelle/Programmiersprachen.

Algorithmische Unlösbarkeit?

prinzipielle Lösbarkeit
↓
effiziente Lösbarkeit
↓
algorithmischer Entwurf
↓
P: Programm in einer HPS

Syntaktische und semantische Verifikation von P.

- Syntaxanalyse
  Sprachen Chomski-Hierarchie
  Kontext freie Sprachen
  Grammatiken/Erzeugungsprozess
- Programmverifikation
  Tut P auch was erwartet wird.

Typische Ausdrücke

- $(x+1)(y-2)/5$ Terme als Bezeichner von Objekten.
- $3+2=5$ Gleichungen als spezielle Formeln.
- „29 ist eine Primzahl“ Aussage.
- „29 ist keine Primzahl“ Aussage.
- „$3+2=5$ und 29 ist keine Primzahl“ Aussage.
- „wenn 29 keine Primzahl ist, dann ist $0=1$“ Aussage.
- „jede gerade Zahl, die größer als 2 ist, ist die Summe zweier Primzahlen“ Aussage.
- $2 \leq x$ und $(\forall y \in \mathbb{N})((2 \leq y$ und $y+1 \leq x) \rightarrow$ nicht$(\exists z \in \mathbb{N}) y * z = x)$ Aussage.
- $(\forall X \subseteq \mathbb{N})(0 \in \mathbb{N} \wedge (\forall x \in \mathbb{N})(x \in X \rightarrow x+1 \in X) \rightarrow X = \mathbb{N})$ Induktionsprinzip.
- $(\forall X \subseteq \mathbb{N})(X \neq \emptyset \rightarrow X$ hat ein kleinstes Element) Jede nichtleere Menge natürlicher Zahlen enthält ein minimales Element.

Zweiwertige Logik: Jede Aussage ist entweder **wahr** oder **falsch**.

Es gibt auch andere Möglichkeiten (Mehrwertige Logik).
Prädikatenlogik erster Stufe (PL1): Nur Eigenschaften von Elementen und Quantifizierung von Elementvariablen erlaubt.

2 Aussagenlogik

- Aussagen $\rightsquigarrow$ Bedeutung **wahr**(1), **falsch** (0)
- Aufbau und Bedeutung von Aussagen. **Syntax** und **Semantik**.

2.1 Syntax

2.1 Definition Die Sprache der Aussagenlogik: Syntax.

Sei $\Sigma = V \cup O \cup K$ Alphabet mit $V = \{p_1, p_2, ...\}$ abzählbare Menge von **Aussagevariablen**, $O = \{\wedge, \vee, \neg, \rightarrow, \leftrightarrow, ...\}$ Menge von **Verknüpfungen** (Junktoren, Operatoren) und $K = \{(,)\}$ **Klammern** Hilfssymbole.

Die Menge der **Aussageformen** (Formeln der Aussagenlogik) $F \subset \Sigma^*$ wird induktiv definiert durch:

1. $V \subset F$ Menge der **atomaren Aussagen**
2. $A, B \in F \quad$ so $(\neg A), (A \wedge B), (A \vee B), (A \rightarrow B), (A \leftrightarrow B) \in F$
3. F ist die kleinste Menge die V enthält und 2. erfüllt (Hüllenoperator)

2.2 Bemerkung

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

Sprache der Aussagenlogik

1. Für $A \in F$ gilt: A ist atomar (ein p_i) oder beginnt mit „(“ und endet mit „)“.
2. Sei $f(A,i) = \sharp$ „(“$-\ \sharp$ „)“ in den ersten i Buchstaben von A, dann gilt $f(A,i) > 0 \quad$ für $1 \leq i < |A|$ und $f(A,i) = 0$ für $i = |A|$.

- F kann als Erzeugnis einer Relation $R \subset U^* \times U$ mit $U = \Sigma^*$ oder eines Kalküls dargestellt werden. Dabei wird F frei von dieser Relation erzeugt, da für alle $u, v \in U^*$ und $A \in F$ gilt: uRA und vRA so $u = v$.
- $F = L(G)$ für eine eindeutige kontextfreie Grammatik G.

2.3 Satz Eindeutigkeitssatz

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

Beweis: Induktion über Aufbau von F.

Vereinbarungen Schreibweisen, Abkürzungen, Prioritäten.

- Äußere Klammern weglassen.
- A-Formen sind z.B.: $p_1, \quad p_{101}, \quad p_1 \vee p_{12}$ als Abkürzung für $(p_1 \vee p_{12})$, $(((p_1 \rightarrow p_2) \wedge (\neg p_2)) \rightarrow (\neg p_1))$, $p_1 \vee (\neg p_1)$

Belegungen und Bewertungen

- Zur besseren Lesbarkeit: Prioritäten $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ d.h.
  $A \wedge B \rightarrow C$ steht für $((A \wedge B) \rightarrow C)$
  $A \vee B \wedge C$ steht für $(A \vee (B \wedge C))$
  $\neg A \vee B \wedge C$ steht für $((\neg A) \vee (B \wedge C))$
  $A \vee B \vee C$ steht für $((A \vee B) \vee C)$ (Linksklammerung).
- Andere Möglichkeiten. „Präfix"- oder „Suffix"- Notation
  Für $(A * B)$ schreibe $*AB$ und für $(\neg A)$ schreibe $\neg A$

2.2 Semantik

2.4 Definition Bewertung, Belegung

Eine **Bewertung** der A-Formen ist eine Funktion $\varphi : F \rightarrow \{0, 1\} = \mathbb{B}$ mit

1. $\varphi(\neg A) = 1 - \varphi(A)$
2. $\varphi(A \vee B) = max(\varphi(A), \varphi(B))$
3. $\varphi(A \wedge B) = min(\varphi(A), \varphi(B))$
4. $\varphi(A \rightarrow B) = \begin{cases} 0 & \text{falls } \varphi(A) = 1 \text{ und } \varphi(B) = 0 \\ 1 & \text{sonst} \end{cases}$
5. $\varphi(A \leftrightarrow B) = \begin{cases} 0 & \text{falls } \varphi(A) \neq \varphi(B) \\ 1 & \text{falls } \varphi(A) = \varphi(B) \end{cases}$

Belegungen und Bewertungen (Fort.)

Sprechweise: A ist „falsch“ unter φ, falls $\varphi(A) = 0$

A ist „wahr“ unter φ oder φ „erfüllt“ A, falls $\varphi(A) = 1$.

Darstellung von Bewertungen durch **Wahrheitstafeln**:

A	$\neg A$
1	0
0	1

A	B	$A \vee B$	$A \wedge B$	$A \rightarrow B$	$A \leftrightarrow B$
0	0	0	0	1	1
0	1	1	0	1	0
1	0	1	0	0	0
1	1	1	1	1	1

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

Offenbar induziert jede Bewertung eine Eindeutige Belegung durch $\psi(p_i) := \varphi(p_i)$.

2.5 Lemma

Jede Belegung $\psi : V \rightarrow \mathbb{B}$ lasst sich auf genau eine Weise zu einer Bewertung $\varphi : F \rightarrow \mathbb{B}$ fortsetzen. Insbesondere wird jede Bewertung durch die Werte auf V eindeutig festgelegt.

Bewertungen

2.6 Folgerung Die Bewertung einer Aussageform $A \in F$ hängt nur von den Werten der in ihr vorkommenden Aussagevariablen aus V ab. Das heißt, will man $\varphi(A)$ berechnen, genügt es, die Werte $\varphi(p)$ zu kennen für alle Aussagevariablen p, die in A vorkommen.

Beispiel:

Sei $\varphi(p) = 1, \varphi(q) = 1, \varphi(r) = 0$. Dann kann $\varphi(A)$ iterativ berechnet werden:

$$A \equiv (\underbrace{(\underbrace{p}_{1} \rightarrow \underbrace{(\underbrace{q}_{1} \rightarrow \underbrace{r}_{0})}_{0})}_{0} \rightarrow \underbrace{(\underbrace{(\underbrace{p}_{1} \wedge \underbrace{q}_{1})}_{1} \rightarrow \underbrace{r}_{0})}_{0})}_{1}$$

Also gilt $\varphi(A) = 1$.

Frage Welche Werte kann $\varphi(A)$ annehmen, wenn φ alle Belegungen durchläuft. Ist etwa $\varphi(A) = 1$ für alle Belegungen φ? Um das nachzuprüfen, „genügt“ es, die endlich vielen unterschiedlichen Belegungen der Variablen, die in A vorkommen, zu überprüfen. Kommen n Variablen in A vor, so gibt es 2^n verschiedene Belegungen. A definiert eine Boolesche Funktion $f_A : \mathbb{B}^n \rightarrow \mathbb{B}$.

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

Bewertungen

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 φ. Weitere solche Formeln sind etwa:

$(A \to (B \to A))$, $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$ oder $((\neg A \to \neg B) \to (B \to A))$.

2.7 Definition Sei $A \in F,\ \Sigma \subseteq F$.

1. (a) A heißt **Tautologie** (**allgemeingültig**), falls $\varphi(A) = 1$ für jede Bewertung φ gilt. (Schreibweise "$\models A$")
   (b) A ist **erfüllbar**, falls es eine Bewertung φ gibt, mit $\varphi(A) = 1$.
   (c) A ist **widerspruchsvoll**, falls $\varphi(A) = 0$ für jede Bewertung φ.
   (d) Schreibe $\mathrm{Taut} = \{A \mid A \in F$ ist Tautologie$\}$, die Menge der Tautologien oder „Theoreme“ der Aussagenlogik, bzw. $\mathrm{Sat} := \{A \mid A \in F$ und A ist erfüllbar$\}$ die Menge der erfüllbaren Formeln.

Semantischer Folgerungsbegriff

2.(a) Σ ist **erfüllbar**, falls es eine Bewertung φ gibt mit $\varphi(A) = 1$ für alle $A \in \Sigma$. ("φ **erfüllt** Σ")

(b) **Semantischer Folgerungsbegriff:** A ist **logische Folgerung** von Σ, falls $\varphi(A) = 1$ für jede Bewertung φ, die Σ erfüllt. Man schreibt "$\Sigma \models A$". Ist $\Sigma = \{A_1, \ldots, A_n\}$, ist die Kurzschreibweise "$A_1, \ldots, A_n \models A$" üblich.

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

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

2.8 Bemerkung Beispiele

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 Folgerung 2.6 sind Tautologien.
   $(p \wedge (\neg p))$ ist widerspruchsvoll. $A \in \mathrm{Taut}$ gdw $\neg A$ widerspruchsvoll
   $(p \wedge q)$ ist erfüllbar jedoch keine Tautologie.
   Die Mengen $\mathrm{Taut}, \mathrm{Sat}$ sind entscheidbar. Beachte $\mathrm{Taut} \subset \mathrm{Sat}$.

2.(a) Sei $\Sigma = \{p\}$ und $A = p \vee q$. Dann gilt $\Sigma \models A$, denn falls $\varphi(p) = 1$, dann auch $\varphi(p \vee q) = 1$. Jede Bewertung, die Σ erfüllt, erfüllt also auch A.

(b) Ist $\Sigma = \emptyset$, dann gilt $\Sigma \models A$ genau dann, wenn A Tautologie ist, d.h. $\mathrm{Folg}(\emptyset) = \mathrm{Taut}$.

(c) Ist Σ nicht erfüllbar, dann gilt $\Sigma \models A$ für alle $A \in F$, d.h. $\mathrm{Folg}(\Sigma) = F$.

Deduktionstheorem und Modus-Ponens Regel

(d) Sei $\Sigma \subseteq \Sigma'$. Ist Σ' erfüllbar, dann ist auch Σ erfüllbar.
(e) Es gilt $\Sigma \subseteq \mathrm{Folg}(\Sigma)$ und $\mathrm{Folg}(\mathrm{Folg}(\Sigma)) = \mathrm{Folg}(\Sigma)$.
(f) Falls $\Sigma \subseteq \Sigma'$, dann gilt $\mathrm{Folg}(\Sigma) \subseteq \mathrm{Folg}(\Sigma')$.

3. $\Sigma \models A$ gilt genau dann, wenn $\Sigma \cup \{\neg A\}$ nicht erfüllbar. Ist Σ endlich, dann ist es entscheidbar, ob Σ erfüllbar ist, und die Menge $\mathrm{Folg}(\Sigma)$ ist entscheidbar.

2.9 Lemma Deduktionstheorem und Modus-Ponens-Regel

a) Deduktionstheorem: $\Sigma, A \models B$ gdw. $\Sigma \models (A \to B)$.
(Σ, A ist Kurzschreibweise für $\Sigma \cup \{A\}$)

b) Modus-Ponens-Regel Es gilt $\{A, A \to B\} \models B$.
Insbesondere ist B eine Tautologie, falls A und $(A \to B)$ Tautologien sind.

Übliche Notationen für Regeln der Form "$A_1, \ldots, A_n \models B$" sind:

$$\begin{array}{c} A_1 \\ \vdots \\ A_n \\ \hline B \end{array} \quad \text{und} \quad \frac{A_1, \ldots, A_n}{B}$$

Für die Modus Ponens Regel also:

$$\frac{A, (A \to B)}{B}\ (\mathbf{MP})$$

Kompaktheitssatz der Aussagenlogik

2.10 Satz Kompaktheitssatz

$\Sigma \subseteq F$ ist erfüllbar genau dann, wenn jede endliche Teilmenge von Σ erfüllbar ist.
Insbesondere gilt $\Sigma \models A$ genau dann, wenn es eine endliche Teilmenge $\Sigma_0 \subseteq \Sigma$ gibt mit $\Sigma_0 \models A$.

2.11 Beispiel

Sei $\Sigma \subseteq F$. Gibt es zu jeder Bewertung φ ein $A \in \Sigma$ mit $\varphi(A) = 1$, so gibt es $A_1, \ldots, A_n \in \Sigma$ $(n > 0)$ mit $\models A_1 \vee \ldots \vee A_n$.

Betrachte die Menge $\Sigma' = \{\neg A \mid A \in \Sigma\}$, nach Voraussetzung ist sie unerfüllbar. Also gibt es eine endliche nichtleere Teilmenge $\{\neg A_1, \ldots, \neg A_n\}$ von Σ' die unerfüllbar ist. Also gilt für jede Bewertung φ gibt es ein i mit $\varphi(\neg A_i) = 0$ oder $\varphi(A_i) = 1$ und somit $\varphi(A_1 \vee \ldots \vee A_n) = 1$.

Der zweite Teil des Satzes ist die Grundlage für Beweisverfahren für $\Sigma \models A$. Dies ist der Fall wenn $\Sigma \cup \{\neg A\}$ unerfüllbar ist. Widerspruchbeweise versuchen systematisch eine endliche Menge $\Sigma_0 \subset \Sigma$ zu finden, so dass $\Sigma_0 \cup \{\neg A\}$ unerfüllbar ist.

Logische Äquivalenz

2.12 Definition Logische Äquivalenz

Seien $A, B \in F$ heißen **logisch äquivalent** mit der Schreibweise $A \models\!=\!\models B$, falls für jede Bewertung φ gilt: $\varphi(A) = \varphi(B)$. Insbesondere ist dann $A \models B$ und $B \models A$.

Einige Beispiele für logisch äquivalente Formeln:

1. $A \models\!=\!\models \neg(\neg A)$,
2. $A \wedge B \models\!=\!\models B \wedge A$ und $A \vee B \models\!=\!\models B \vee A$,
3. $A \wedge (B \wedge C) \models\!=\!\models (A \wedge B) \wedge C$ und $A \vee (B \vee C) \models\!=\!\models (A \vee B) \vee C$,
4. $A \wedge (B \vee C) \models\!=\!\models (A \wedge B) \vee (A \wedge C)$ und $A \vee (B \wedge C) \models\!=\!\models (A \vee B) \wedge (A \vee C)$ (**De Morgan**)
5. $\neg(A \wedge B) \models\!=\!\models (\neg A) \vee (\neg B)$ und $\neg(A \vee B) \models\!=\!\models (\neg A) \wedge (\neg B)$
6. $A \to B \models\!=\!\models (\neg A) \vee B$, $A \wedge B \models\!=\!\models \neg(A \to (\neg B))$ und $A \vee B \models\!=\!\models (\neg A) \to B$.
7. $A \leftrightarrow B \models\!=\!\models (A \to B) \wedge (B \to A)$

Man beachte, dass $\models\!=\!\models$ reflexiv, transitiv und symmetrisch, d.h. eine Äquivalenzrelation ist.

Logische Äquivalenz (Fort.)

Folgende Aussagen sind äquivalent:

- $\models (A \leftrightarrow B)$
- $A \models B$ und $B \models A$
- $A \models\!=\!\models B$
- $\mathrm{Folg}(A) = \mathrm{Folg}(B)$

2.13 Folgerung

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

1. $A \models\!=\!\models B$, B enthält nur $\longrightarrow$ und $\neg$ als log. Verknüpfungen
2. $A \models\!=\!\models C$, C enthält nur $\wedge$ und $\neg$ als log. Verknüpfungen
3. $A \models\!=\!\models D$, D enthält nur $\vee$ und $\neg$ als log. Verknüpfungen

2.14 Definition Vollständige Operatorenmengen

Eine Menge $OP \subseteq \{\neg, \vee, \wedge, \longrightarrow, \leftrightarrow, ..\}$ heißt **vollständig**, falls es zu jedem $A \in F$ eine logisch äquivalente A-Form $B \in F(OP)$ gibt.

Vollständige Operatorenmengen für die Aussagenlogik sind z.B.:

$\{\neg, \longrightarrow\}, \{\neg, \vee\}, \{\neg, \wedge\}, \{\neg, \vee, \wedge\}, \{\text{false}, \longrightarrow\}$

Dabei ist false eine Konstante, mit $\varphi(\text{false}) = 0$ für jede Bewertung φ. Offenbar gilt $\neg A \models\!=\!\models (A \longrightarrow \text{false})$.

Normalformen DNF (Disjunktive Normalform), KNF (Konjunktive Normalform), KDNF, KKNF (Kanonische Formen).

Boolsche Funktionen

Jede Aussageform $A(p_1, \ldots, p_n)$ stellt in natürlicher Form eine Boolsche Funktion $f_A : \mathbb{B}^n \longrightarrow \mathbb{B}$ dar. Nämlich durch die Festlegung

$f_A(b_1, \ldots, b_n) = \varphi_{\vec{b}}(A)$ mit der Bewertung $\varphi_{\vec{b}}(p_i) = b_i$

Man kann leicht durch Induktion nach n zeigen, dass jede Boolsche Funktion $f : \mathbb{B}^n \longrightarrow \mathbb{B}$ $(n > 0)$ sich in obiger Form durch eine Aussageform in $p_1, \ldots, p_n$ und einer vollständigen Operatorenmenge darstellen lässt.

Die Boolsche Algebra über $\mathbb{B}$ hat als übliche Operatormenge **true, false, not, or, and** mit der standard Interpretation.

Für andere Operatormengen die etwa **nand, nor** enthalten, siehe Digitale Logik (Gatter: Ein- Ausgabesignale, Verzögerung. **nand, nor** Gattern werden bevorzugt, da nur zwei Transistoren nötig).

Beispiel Patientenüberwachungssystem erhält gewisse Daten über den Zustand eines Patienten. Z.B. Temperatur, Blutdruck, Pulsrate. Die Schwellenwerte für die Daten seien etwa 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 min.

O Alarmaktivierung ist notwendig.

Beispiel Patientenüberwachungssystem

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 die O den Wert 1 hat und stelle eine KDNF auf (Disjunktion von Konjunktionen von Literalen, wobei ein Literal eine atomare Form oder die Negation einer solchen ist).

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

Als eine Realisierung könnte man das folgende Schaltnetz nehmen: