Was ist Logik?

Logik (nach dem Altgriechischen „Logos“: „Vernunft“) ist

die Lehre des vernünftigen Schlussfolgerns.

Die für die Logik zentrale Frage:

Wie kann man Aussagen miteinander verknüpfen, und auf welche Weise kann man formal Schlüsse ziehen und Beweise führen?

Logik in der Informatik

Logik wird in der Informatik u.a. genutzt

- für die **Beschreibung, Analyse, Optimierung** und **Verifikation** digitaler Schaltungen,
- für den **Nachweis**, dass ein Programm gewisse wünschenswerte Eigenschaften hat,
- für die **Wissensrepräsentation**, z.B. im Bereich der künstlichen Intelligenz,
- als Grundlage für **Datenbank-Anfragesprachen**,
- für die **automatische Erzeugung von Beweisen** in so genannten „Theorembeweisern“.

Aussagenlogik

Wir beschäftigen uns mit **Aussagen**.

- Aussagen im Sinne der Aussagenlogik sind sprachliche Gebilde, die entweder **wahr** oder **falsch** sind.

  *Aussagen können mit **Junktoren** wie „nicht“, „und“, „oder“, „wenn . . . dann“ etc. zu komplexeren Aussagen verknüpft werden.*

- Die **Aussagenlogik** beschäftigt sich

  mit allgemeinen Prinzipien des korrekten Argumentierens und Schließens mit Aussagen und Kombinationen von Aussagen.

Syntax und Semantik der Aussagenlogik

Formeln und ihre Bedeutung

- Die **Syntax** legt fest, welche Zeichenketten Formeln der Aussagenlogik sind.
- Die **Semantik** legt fest, welche „Bedeutung“ die einzelnen Formeln haben.

Für Programmiersprachen ist die Situation ähnlich:

- Die Syntax legt fest, welche Zeichenketten korrekte Programme sind,
- während die Semantik bestimmt, was das Programm tut.

Die Syntax der Aussagenlogik

Aussagenvariablen

DEFINITION

(a) Eine **Aussagenvariable** (kurz: Variable) hat die Form V_i für $i \in \mathbb{N}$.
Die Menge aller Aussagenvariablen bezeichnen wir mit AVAR. D.h.:

$$\text{AVAR} \;:=\; \{V_i \;:\; i \in \mathbb{N}\} \;=\; \{V_0, V_1, V_2, V_3, \ldots\}.$$

(b) Das **Alphabet** A_{AL} der Aussagenlogik ist

$$A_{\text{AL}} \;:=\; \text{AVAR} \cup \{\mathbf{0}, \mathbf{1}, \neg, \wedge, \vee, \rightarrow, \leftrightarrow, \oplus, (,)\}$$

A_{AL} besteht aus den Aussagenvariablen, den aussagenlogischen Variablen $\mathbf{0}, \mathbf{1}$, den Junktoren $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \oplus$ und den runden Klammern.

Was sind aussagenlogische Formeln?

Aussagenlogische Formeln

DEFINITION

Die Menge AL der aussagenlogischen Formeln wird **rekursiv** definiert:

Basisregeln:

(B0) $\mathbf{0} \in \mathrm{AL}$.

(B1) $\mathbf{1} \in \mathrm{AL}$.

(BV) Für jede Variable $V \in \mathrm{AVAR}$ gilt: $V \in \mathrm{AL}$.

Rekursive Regeln:

(R1) Ist $\phi \in \mathrm{AL}$, so ist auch $\neg\phi \in \mathrm{AL}$.

(R2) Ist $\phi \in \mathrm{AL}$ und $\psi \in \mathrm{AL}$, so ist auch

- $(\phi \wedge \psi) \in \mathrm{AL}$
- $(\phi \vee \psi) \in \mathrm{AL}$
- $(\phi \rightarrow \psi) \in \mathrm{AL}$
- $(\phi \leftrightarrow \psi) \in \mathrm{AL}$
- $(\phi \oplus \psi) \in \mathrm{AL}$

Eine Formel ϕ gehört genau dann zu AL, wenn man ϕ durch (möglicherweise mehrfache) Anwendung der obigen Regeln erzeugen kann.

Beispiele

- $\big(\neg V_0 \vee (V_5 \to V_1)\big)$ ist eine korrekt gebildete Formel.
- $V_1 \vee V_2 \vee V_3$ ist keine korrekt gebildete Formel, da die Klammern fehlen,
  - In diesem Fall bleibt die Bedeutung aber klar (warum?).
- $\neg\big((V_0 \wedge \mathbf{0}) \leftrightarrow \neg V_3\big)$ ist hingegen wieder eine korrekte Formel,
- $(\neg V_1)$ ist keine korrekte Formel, da sie zu viele Klammern besitzt.
  - Aber auch in diesem Fall bleibt die Bedeutung klar.
- Aber was soll $V_0 \wedge V_1 \vee V_2$ bedeuten?
  - Möglicherweise $(V_0 \wedge V_1) \vee V_2$
  - oder $V_0 \wedge (V_1 \vee V_2)$?

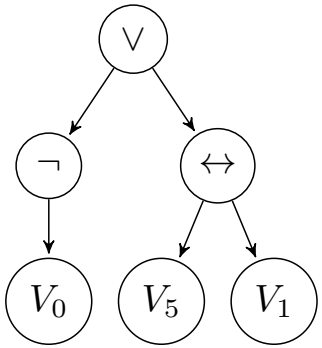
Übrigens, von welcher Bedeutung reden wir überhaupt?

Der Syntaxbaum einer Formel

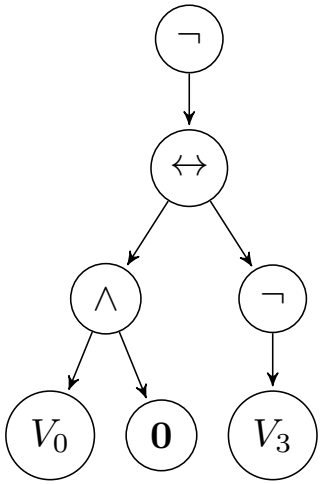
Die Struktur einer Formel lässt sich durch einen **Syntaxbaum** darstellen.

Beispiele:

- Syntaxbaum der Formel $\big(\neg V_0 \vee (V_5 \leftrightarrow V_1)\big)$:



- Syntaxbaum der Formel $\neg\big((V_0 \wedge \mathbf{0}) \leftrightarrow \neg V_3\big)$:



Bezeichnungen

(a) **0** (stets falsch), **1** (stets wahr) und die Variablen (d.h. die Elemente aus AVAR) bezeichnen wir als **atomare Formeln** bzw. **Atome**.

(b) Die Symbole $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\oplus$ heißen **Junktoren**.

(c) Sind ϕ und ψ Formeln (d.h. $\phi \in$ AL und $\psi \in$ AL), so heißt:

- $\neg\phi$ **Negation** (bzw. Ver*nein*ung) von ϕ,
- $(\phi \wedge \psi)$ **Konjunktion** (bzw. Ver*und*ung) von ϕ und ψ,
- $(\phi \vee \psi)$ **Disjunktion** (bzw. Ver*oder*ung) von ϕ und ψ,
- $(\phi \rightarrow \psi)$ **Implikation** (bzw. Subjunktion) von ϕ und ψ,
- $(\phi \leftrightarrow \psi)$ **Biimplikation** (bzw. Äquivalenz oder Bijunktion) von ϕ und ψ, und
- $(\phi \oplus \psi)$ **Xor** (bzw. exklusives Oder) von ϕ und ψ.

Das Leben ist schon so hart genug!

- Statt $V_0, V_1, V_2, \ldots$ bezeichnen wir Variablen oft auch mit $A, B, C, \ldots, X, Y, Z, \ldots$ oder mit Variablen wie $X', Y_1, \ldots$
- Die äußeren Klammern einer Formel lassen wir manchmal weg und schreiben z.B. $(A \wedge B) \to C$ an Stelle des (formal korrekten) $\big((A \wedge B) \to C\big)$.

Wenn Sie eine Formel „vereinfachen“, stellen Sie sicher, dass

die Bedeutung eindeutig ist!

Aussagenlogische Formeln: Beispiele

Geburtstagsfeier (1/3)

Fred möchte seinen Geburtstag mit möglichst vielen seiner Freunde,

nämlich mit Anne, Bernd, Christine, Dirk und Eva

feiern.

- Er weiß, dass Eva nur dann kommt, wenn Christine und Dirk kommen.
- Andererseits kommt Christine nur dann, wenn auch Anne kommt;
- und Dirk wird auf keinen Fall kommen,
  wenn Bernd und Eva beide zur Feier kommen.
- Anne wiederum wird nur dann kommen,
  wenn auch Bernd oder Christine dabei sind.
- Wenn allerdings Bernd und Anne beide zur Party kommen,
  dann wird Eva auf keinen Fall dabei sein.

Wie viele Freunde, und welche, werden im besten Fall zur Party kommen?

Geburtstagsfeier (2/3)

Das Wissen, das im obigen Text wiedergegeben ist, lässt sich in „atomare Aussagen“ zerlegen, die mit Junktoren verknüpft werden können.

Um welche “atomaren Aussagen“ dreht sich der Text?

$$
\begin{array}{lcl}
A & \widehat{=} & \text{Anne kommt zur Feier} \\
B & \widehat{=} & \text{Bernd kommt zur Feier} \\
C & \widehat{=} & \text{Christine kommt zur Feier} \\
D & \widehat{=} & \text{Dirk kommt zur Feier} \\
E & \widehat{=} & \text{Eva kommt zur Feier.}
\end{array}
$$

Geburtagsfeier (3/3)

Das im Text zusammengefasste „Wissen“ lässt sich wie folgt repräsentieren:

1. Eva kommt nur dann, wenn Christine und Dirk kommen.
   - $\phi_1 := (E \to (C \wedge D))$
2. Christine kommt nur dann, wenn auch Anne kommt,
   - und $\phi_2 := (C \to A)$
3. Dirk wird auf keinen Fall kommen, wenn Bernd und Eva beide kommen,
   - und $\phi_3 := ((B \wedge E) \to \neg D)$
4. Anne kommt nur dann, wenn auch Bernd oder Christine dabei sind,
   - und $\phi_4 := (A \to (B \vee C))$
5. wenn Bernd und Anne kommen, dann wird Eva auf keinen Fall dabei sein.
   - und $\phi_5 := ((B \wedge A) \to \neg E)$.

Die vollständige Wissensrepräsentation ist die Konjunktion

$$\psi := (\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4 \wedge \phi_5)$$

Mit der Aussagenlogik können wir „Wissen“ modellieren und Schlüsse daraus ziehen: Aber wieviele Freunde kommen denn nun bestenfalls?

Wir modellieren Zeugenaussagen

Wir möchten die Zeugenaussage

> *„Das Fluchtauto war rot oder grün und*
> *hatte weder vorne noch hinten ein Nummernschild.“*

durch eine aussagenlogische Formel repräsentieren. Dazu verwenden wir die folgenden atomaren Aussagen:

- X_R: das Fluchtauto war rot,
- X_G: das Fluchtauto war grün,
- X_V: das Fluchtauto hatte vorne ein Nummernschild,
- X_H: das Fluchtauto hatte hinten ein Nummernschild.

Wir repräsentieren die Zeugenaussage durch

$$\big((X_R \oplus X_G) \wedge (\neg X_V \wedge \neg X_H)\big).$$

Die Semantik der Aussagenlogik

Die Variablen einer Formel

DEFINITION

Die **Variablenmenge**

$$\mathrm{Var}(\phi)$$

einer aussagenlogischen Formel ϕ ist die Menge aller Variablen $X \in \mathrm{AVAR}$, die in ϕ vorkommen.

- $\mathrm{Var}\Big(\big(\neg V_0 \vee (V_5 \to V_1)\big)\Big) = \{V_0, V_1, V_5\}$,
- $\mathrm{Var}\Big(\neg\big((V_0 \wedge \mathbf{0}) \leftrightarrow \neg V_3\big)\Big) = \{V_0, V_3\}$,
- $\mathrm{Var}\Big((\mathbf{0} \vee \mathbf{1})\Big) = \emptyset$.

Belegungen

Belegung = Zuweisung von Wahrheitswerten zu Variablen

DEFINITION

(a) Eine **partielle** Funktion

$$\mathcal{B} : \text{AVAR} \to \{0, 1\}$$

von AVAR nach $\{0, 1\}$ heißt eine

Belegung,

bzw. Wahrheitsbelegung.

- 1 steht für den Wert „wahr“ und 0 für den Wert „falsch“.

(b) Eine Belegung $\mathcal{B}$ ist eine Belegung für die Formel ϕ (bzw. **passend** zu ϕ), wenn $\mathcal{B}$ auf allen Variablen von ϕ definiert ist.

Ein Beispiel: Die Funktion $\mathcal{B}$

mit $\mathcal{B}(V_0) = 1, \mathcal{B}(V_1) = 1$ und $\mathcal{B}(V_3) = 0$

besitzt den Definitionsbereich $\{V_0, V_1, V_3\}$. $\mathcal{B}$ ist eine Belegung für $\big(V_0 \wedge (V_1 \vee V_3)\big)$ oder $(V_0 \wedge V_3)$, nicht aber für $(V_0 \wedge V_2)$.

Wann ist eine Formel für eine Belegung wahr?

Der Wahrheitswert einer Formel

Wir definieren eine Funktion $[\![\phi]\!]^{\mathcal{B}}$,

- die jeder Formel $\phi \in \mathrm{AL}$
- und jeder zu ϕ passenden Belegung $\mathcal{B}$

einen **Wahrheitswert** (kurz: Wert) $[\![\phi]\!]^{\mathcal{B}} \in \{0, 1\}$ zuordnet.

Wir benutzen wieder, wie im Fall der Definition der Syntax,

eine rekursive Definition über den Aufbau aussagenlogischer Formeln.

1. Zuerst definieren wir Wahrheitswerte für atomare Formeln
2. und dann für Negationen, Konjunktionen, Disjunktionen, Implikationen, Äquivalenzen und XOR.

Der Wahrheitswert einer Formel: Die Definition

Rekursionsanfang: $[\![\mathbf{0}]\!]^{\mathcal{B}} := 0$, $[\![\mathbf{1}]\!]^{\mathcal{B}} := 1$ und
F.a. $X \in \mathrm{AVAR}$, für die $\mathcal{B}$ definiert ist, gilt: $[\![X]\!]^{\mathcal{B}} := \mathcal{B}(X)$.

Rekursionsschritt:

- Ist $\phi \in \mathrm{AL}$, so ist $[\![\neg\phi]\!]^{\mathcal{B}} := \begin{cases} 1, & \text{falls } [\![\phi]\!]^{\mathcal{B}} = 0 \\ 0, & \text{falls } [\![\phi]\!]^{\mathcal{B}} = 1. \end{cases}$
- Ist $\phi \in \mathrm{AL}$ und $\psi \in \mathrm{AL}$, so ist
  - $[\![(\phi \wedge \psi)]\!]^{\mathcal{B}} := \begin{cases} 1, & \text{falls } [\![\phi]\!]^{\mathcal{B}} = 1 \text{ und } [\![\psi]\!]^{\mathcal{B}} = 1 \\ 0, & \text{sonst} \end{cases}$
  - $[\![(\phi \vee \psi)]\!]^{\mathcal{B}} := \begin{cases} 0, & \text{falls } [\![\phi]\!]^{\mathcal{B}} = 0 \text{ und } [\![\psi]\!]^{\mathcal{B}} = 0 \\ 1, & \text{sonst} \end{cases}$
  - $[\![(\phi \to \psi)]\!]^{\mathcal{B}} := \begin{cases} 0, & \text{falls } [\![\phi]\!]^{\mathcal{B}} = 1 \text{ und } [\![\psi]\!]^{\mathcal{B}} = 0 \\ 1, & \text{sonst} \end{cases}$
  - $[\![(\phi \leftrightarrow \psi)]\!]^{\mathcal{B}} := \begin{cases} 1, & \text{falls } [\![\phi]\!]^{\mathcal{B}} = [\![\psi]\!]^{\mathcal{B}} \\ 0, & \text{sonst.} \end{cases}$
  - $[\![(\phi \oplus \psi)]\!]^{\mathcal{B}} := \begin{cases} 1, & \text{falls } [\![\phi]\!]^{\mathcal{B}} \neq [\![\psi]\!]^{\mathcal{B}} \\ 0, & \text{sonst.} \end{cases}$

Widersprüche und Tautologien

Erfüllende und falsifizierende Belegungen

ϕ sei eine Formel und $\mathcal{B}$ eine Belegung für ϕ.

DEFINITION

1. $\mathcal{B}$ **erfüllt** ϕ (bzw. $\mathcal{B}$ ist eine erfüllende Belegung für ϕ), falls

$$[\![\phi]\!]^{\mathcal{B}} = 1.$$

2. $\mathcal{B}$ **falsifiziert** ϕ (bzw. $\mathcal{B}$ ist eine falsifizierende Belegung für ϕ), falls

$$[\![\phi]\!]^{\mathcal{B}} = 0.$$

Erfüllbare, widersprüchliche und allgemeingültige Formeln

DEFINITION

Sei ϕ eine aussagenlogische Formel.

(a) ϕ heißt **erfüllbar**, wenn es **mindestens eine** erfüllende Belegung $\mathcal{B}$ für ϕ gibt.

- Die Belegung $\mathcal{B}$ passt zu ϕ und es gilt $[\![\phi]\!]^{\mathcal{B}} = 1$.

(b) ϕ heißt **falsifizierbar**, wenn es **mindestens** eine falsifizierende Belegung für ϕ gibt,

- Die Belegung $\mathcal{B}$ passt zu ϕ und es gilt $[\![\phi]\!]^{\mathcal{B}} = 0$.

(c) ϕ heißt **unerfüllbar** (oder **widersprüchlich**), wenn **alle** zu ϕ passenden Belegungen ϕ falsifizieren.

(d) ϕ heißt **allgemeingültig** (oder eine **Tautologie**), wenn **alle** zu ϕ passenden Belegungen ϕ erfüllen.

Beispiele

(a) Die Formel

$$((X \vee Y) \wedge (\neg X \vee Y))$$

ist **erfüllbar** und deshalb **nicht widerspüchlich**,

- denn jede Belegung $\mathcal{B}$ mit $\mathcal{B}(Y) = 1$ erfüllt die Formel,

aber auch **falsifizierbar** und deshalb **nicht allgemeingültig**,

- denn jede Belegung $\mathcal{B}'$ mit $\mathcal{B}'(Y) = 0$ falsifiziert die Formel.

(b) Die Formel

$$(X \wedge \neg X)$$

ist **unerfüllbar** (oder **widersprüchlich**), da für jede zur Formel passenden Belegung $\mathcal{B}$ entweder $\mathcal{B}(X) = 1$ oder $\mathcal{B}(X) = 0$ gilt.

- Ist $\mathcal{B}(X) = 1$, so gilt: $[\![(X \wedge \neg X)]\!]^{\mathcal{B}} = 1 \wedge 0 = 0$,
- Ist $\mathcal{B}(X) = 0$, so gilt: $[\![(X \wedge \neg X)]\!]^{\mathcal{B}} = 0 \wedge 1 = 0$.

(c) Die Formel

$$(X \vee \neg X)$$

ist **allgemeingültig**, da für jede zur Formel passenden Belegung $\mathcal{B}$ entweder $\mathcal{B}(X) = 1$ oder $\mathcal{B}(\neg X) = 1$ gilt.

Wie behält man den Überblick über alle Belegungen?

Zum Beispiel mit **Wahrheitstafeln**!

Wahrheitstafel (1/2)

Für jede Formel ψ kann man

die Wahrheitswerte von ψ unter allen möglichen Belegungen

in einer Wahrheitstafel (manchmal auch Funktionstafel oder Wahrheitstabelle genannt) darstellen.

DEFINITION

(a) Die **Wahrheitstafel** hat für jede Belegung $\mathcal{B} : \mathrm{Var}(\psi) \to \{0, 1\}$ eine Zeile.

(b) Die Zeile für $\mathcal{B}$ enthält mindestens

1. f.a. $X \in \mathrm{Var}(\psi)$ die Werte $\mathcal{B}(X)$ und
2. den Wert $[\![\psi]\!]^{\mathcal{B}}$.

Um die Wahrheitstafel für ϕ fehlerfrei auszufüllen, ist es ratsam, auch Spalten für (alle oder einige) „Teilformeln“ von ϕ einzufügen.

Wahrheitstafel (2/2)

(a) Wahrheitstafel für $\phi := \Big(X \wedge \big((\mathbf{1} \to \mathbf{0}) \to \mathbf{0}\big)\Big)$

X	$\mathbf{1}$	$\mathbf{0}$	$(\mathbf{1} \to \mathbf{0})$	$((\mathbf{1} \to \mathbf{0}) \to \mathbf{0})$	ϕ
0	1	0	0	1	0
1	1	0	0	1	1

Also ist ϕ sowohl erfüllbar wie auch falsifizierbar.

(b) Wahrheitstafel für $\psi := \Big((A \to B) \vee (A \wedge \neg B)\Big)$

A	B	$A \to B$	$A \wedge \neg B$	ψ
0	0	1	0	1
0	1	1	0	1
1	0	0	1	1
1	1	1	0	1

Also ist ψ allgemeingültig.

Wahrheitstafeln für die Junktoren (1/4)

- **Atomare Formeln**:
  - **1** und **0** bedeuten einfach „wahr“ und „falsch“.
  - Die Variablen $X \in \mathrm{AVAR}$ stehen für irgendwelche Aussagen.
    Uns interessiert hier nur, ob diese Aussagen „wahr“ oder „falsch“ sind — und dies wird durch eine Belegung $\mathcal{B}$ angegeben.
- **Negation**: $\neg\phi$ bedeutet „*nicht* ϕ“.

  Zugehörige Wahrheitstafel:

$[\![\phi]\!]^{\mathcal{B}}$	$[\![\neg\phi]\!]^{\mathcal{B}}$
0	1
1	0

Wahrheitstafeln für Junktoren (2/4)

- **Konjunktion**: $(\phi \wedge \psi)$ bedeutet „ϕ *und* ψ“.

  Zugehörige Wahrheitstafel:

$[\![\phi]\!]^{\mathcal{B}}$	$[\![\psi]\!]^{\mathcal{B}}$	$[\![(\phi \wedge \psi)]\!]^{\mathcal{B}}$
0	0	0
0	1	0
1	0	0
1	1	1

- **Disjunktion, bzw. inklusives Oder**: $(\phi \vee \psi)$ bedeutet „ϕ *oder* ψ“.

  Zugehörige Wahrheitstafel:

$[\![\phi]\!]^{\mathcal{B}}$	$[\![\psi]\!]^{\mathcal{B}}$	$[\![(\phi \vee \psi)]\!]^{\mathcal{B}}$
0	0	0
0	1	1
1	0	1
1	1	1

Wahrheitstafeln für die Junktoren (3/4)

- **Implikation**: $(\phi \to \psi)$ bedeutet „ϕ *impliziert* ψ“, d.h. „*wenn* ϕ, *dann auch* ψ“, bzw. „ϕ *nur wenn* ψ“.

  Zugehörige Wahrheitstafel:

$[\![\phi]\!]^{\mathcal{B}}$	$[\![\psi]\!]^{\mathcal{B}}$	$[\![(\phi \to \psi)]\!]^{\mathcal{B}}$
0	0	1
0	1	1
1	0	0
1	1	1

- **Äquivalenz**: $(\phi \leftrightarrow \psi)$ bedeutet „ϕ *gilt genau dann, wenn* ψ gilt“.

  Zugehörige Wahrheitstafel:

$[\![\phi]\!]^{\mathcal{B}}$	$[\![\psi]\!]^{\mathcal{B}}$	$[\![(\phi \leftrightarrow \psi)]\!]^{\mathcal{B}}$
0	0	1
0	1	0
1	0	0
1	1	1

Wahrheitstafeln für die Junktoren (4/4)

- **Xor, bzw. exklusives Oder**: $(\phi \oplus \psi)$ bedeutet „entweder ϕ *oder* ψ“, bzw. „genau eine der beiden Aussagen ϕ, ψ“

  Zugehörige Wahrheitstafel:

$[\![\phi]\!]^{\mathcal{B}}$	$[\![\psi]\!]^{\mathcal{B}}$	$[\![(\phi \oplus \psi)]\!]^{\mathcal{B}}$
0	0	0
0	1	1
1	0	1
1	1	0

Der Sprachgebrauch für „oder“ ist unscharf und meint

- manchmal das „inklusive Oder“
  (wie in „Anna oder Xaver helfen beim Umzug“) und
- manchmal das „exklusive Oder“
  (wie in „Ansgar oder Xenia fahren den Laster“).

Und nochmals Wahrheitstafeln

Für jede aussagenlogische Formel ϕ gilt:

(a) ϕ ist erfüllbar $\iff$ in der Wahrheitstafel für ϕ steht in der mit „ϕ“ beschrifteten Spalte mindestens eine 1.

(b) ϕ ist unerfüllbar $\iff$ in der Wahrheitstafel für ϕ stehen in der mit „ϕ“ beschrifteten Spalte nur Nullen.

(c) ϕ ist allgemeingültig $\iff$ in der Wahrheitstafel für ϕ stehen in der mit „ϕ“ beschrifteten Spalte nur Einsen.

(d) ϕ ist allgemeingültig $\iff$ $\neg\phi$ ist unerfüllbar.

(e) ϕ ist falsifizierbar $\iff$ in der Wahrheitstafel für ϕ steht in der mit „ϕ“ beschrifteten Spalte mindestens eine 0.

$\iff$ ϕ ist nicht allgemeingültig.

Kommutativität und Assoziativität

Die Junktoren

$$\vee, \wedge, \leftrightarrow, \oplus$$

sind kommutativ und assoziativ.

D.h. für $\circ \in \{ \vee, \wedge, \leftrightarrow, \oplus \}$, für alle Belegungen $\mathcal{B}$ und für alle aussagenlogischen Formeln ϕ, χ und ψ gilt:

- **Kommutativität**: $\phi \circ \psi$ ist genau dann wahr, wenn $\psi \circ \phi$ wahr ist.
- **Assoziativität**: $\phi \circ (\chi \circ \psi)$ ist genau dann wahr, wenn $(\phi \circ \chi) \circ \psi$ wahr ist. Der Wahrheitswert hängt also nicht von der Klammerung ab.

  Wir schreiben deshalb $\bigwedge_{i=1}^{n} \phi_i$ bzw. $\phi_1 \wedge \cdots \wedge \phi_n$ an Stelle von $\Big(\big((\phi_1 \wedge \phi_2) \wedge \phi_3 \big) \wedge \cdots \wedge \phi_n \Big)$ Analog für „$\vee, \leftrightarrow$ und $\oplus$“ an Stelle von „$\wedge$“.

Achtung: Die Implikation $\rightarrow$ ist weder kommutativ noch assoziativ.

Frage: Für welche Belegungen ist $V_1 \circ \cdots \circ V_n$ wahr, wenn $\circ \in \{ \vee, \wedge, \leftrightarrow, \oplus \}$?

Aussagenlogische Formeln: Python

Python: Aussagenlogik und der Datentyp `bool`

Python-Objekte vom Datentyp `bool` modellieren Aussagenvariablen.
Die Funktion `bool()` gibt Objekte vom Datentyp `bool` aus: Das Objekt x in

```
x = bool(y)
```

hat also den Datentyp `bool`, und es ist

```
bool ({})   =  bool (()) = bool ([]) = bool (‘‘)
            =  ···= bool (0) = False
```

Für „alle anderen“ Argumente nimmt `bool` den Wert `True` an.

Wie sehen Junktoren in Python aus?

Python: Junktoren und Python-Operatoren

Die folgenden Python-Operatoren entsprechen Junktoren:

1. `not x` entspricht der Negation $\neg x$,
2. `x and y`, bzw. das bitweise-Und `x & y` entspricht der Konjunktion $x \wedge y$,
3. `x or y`, bzw. das bitweise-Oder `x | y` entspricht der Disjunktion $x \vee y$,
4. der „Vergleich“ `x <= y` entspricht der Implikation $x \rightarrow y$,
5. der Gleichheitstest `x == y` entspricht der Äquivalenz $x \leftrightarrow y$,
6. die Ungleichheit `x != y` entspricht dem Xor $x \oplus y$.

Python: Aussagenlogische Formeln

+ Mit Hilfe dieser Python-Operatoren und der runden Klammern können aussagenlogische Python-Formeln ϕ „gebaut“ werden.
- Allerdings versucht Python sofort, ϕ zu evaluieren und meldet einen Fehler, wenn dies nicht gelingt – z.B. weil einige Variablen nicht definiert wurden –

Für die Definition aussagenlogischer Formeln – ohne unmittelbare Evaluierung – arbeiten wir mit **SymPy**.

SymPy und Erfüllbarkeit

SymPy

SymPy ist eine Python-Bibliothek für Aufgaben in der symbolischen Mathematik.

Wir benutzen die „App“ ***SymPy Live*** (`http://live.sympy.org/`):
Aussagenlogische Formeln können in einer Shell definiert und untersucht werden.

1. Aussagenlogische Variable wie etwa `u, v, mu` werden deklariert mit

```
u,v,mu = symbols(‘u,v,mu‘)
```

   - Standardmäßig werden `x,y,z,t` in SymPy Live als Variablennamen deklariert.

2. **SymPy-Formeln**:
   - Atomare SymPy-Formeln sind die aussagenlogischen Konstanten `True, False` und die deklarierten aussagenlogischen Variablen.
   - Man erhält alle SymPy-Formeln, wenn bereits konstruierte SymPy-Formeln geklammert oder mit „**SymPy-Junktoren**“ verbunden werden.

Die SymPy-Junktoren

$x, y, x1, \ldots, xk$ dürfen durch bereits konstruierte SymPy-Formeln ersetzt werden.

(a) `Not(x)` entspricht der Negation $\neg x$,

(b) `And(x1, ..., xk)` bzw. das „bitweise-Und“ `x1 &...& xk` entspricht der Konjunktion $x1 \wedge \cdots \wedge xk$,

(c) `Or(x1, ..., xk)` bzw. das „bitweise-Oder“ `x1 |...| xk` entspricht der Disjunktion $x1 \vee \cdots \vee xk$,

(d) `Implies(x,y)` entspricht der Implikation $x \rightarrow y$,

(e) `Equivalent(x1, ...,xk)` entspricht der Äquivalenz $x1 \leftrightarrow \cdots \leftrightarrow xk$,

(f) und `Xor(x1, ..., xk)` bzw. `x1 ^...^ xk` entspricht dem Xor $x1 \oplus \cdots \oplus xk$.

SymPy: Eingabe von Formeln

Formeln können mit der Zuweisung (=) gespeichert werden.

Aufgabe: Schreibe die aussagenlogische Formel

$$(x \oplus y) \leftrightarrow ((x \wedge \neg y) \vee (\neg x \wedge y))$$

als SymPy-Formel `mu` in SymPy Live. Zum Beispiel als

```
>>> phi = (x ^ y)
>>> psi = ((x & Not(y)) | (Not(x) & y))
>>> mu = Equivalent(phi,psi)
```

oder mit

```
>>> phi = Xor(x, y)
>>> psi = Or( And(x, Not(y)) , And(Not(x),y))
>>> mu = Equivalent(phi,psi)
```

oder in „einem Schlag“

```
>>> mu = Equivalent( Xor(x, y) ,
... Or( And(x, Not(y)) , And(Not(x),y)) )
```

SymPy: Die Funktion `satisfiable` (1/2)

`mu` sei eine SymPy-Formel.

Wir können die Funktion `satisfiable()` mit `mu` als Argument aufrufen:

- Ist `mu` erfüllbar, wird SymPy irgendeine erfüllende Belegung angeben,
- ist `mu` unerfüllbar, antwortet SymPy mit `False`.

Mit der Formel `mu` aus der letzten Folie erhalten wir

```
>>> satisfiable( mu )
x:False, y:False
```

bzw.

```
>>> satisfiable( Not(mu) )
False
```

SymPy: Die Funktion `satisfiable` (2/2)

Wie gibt man alle erfüllenden Belegungen einer Formel `mu` aus?

Wir schreiben eine Funktion

```
>>> def erfuellen (models):
....  for model in models:
....         if model:
....              print(model)
....         else:
....              print('Die Formel ist unerfüllbar')
```

Der Aufruf

```
>>> erfuellen( satisfiable (mu, all_models=True))
```

gibt alle erfüllenden Belegungen aus bzw. erkennt dass die Formel unerfüllbar ist.
Den gleichen Effekt hat: `>>> list( satisfiable (mu, all_models=True))`

Semantische Folgerung und Äquivalenz

(Semantische) Folgerung

ϕ, ψ seien aussagenlogische Formeln und Φ eine Menge von Formeln.

DEFINITION

1. ψ **folgt aus** ϕ, bzw. ϕ **impliziert** ψ, kurz

$$\phi \models \psi,$$

   falls $\phi \to \psi$ allgemeingültig ist (d.h. falls $[\![\phi \to \psi]\!]^{\mathcal{B}} = 1$ für **jede** zu ϕ und ψ passende Belegung $\mathcal{B}$ gilt).

2. ψ **folgt aus** Φ, bzw. Φ **impliziert** ψ, kurz

$$\Phi \models \psi,$$

   falls **jede** Belegung, die zu allen Formeln in Φ und zu ψ passt und die alle Formeln in Φ erfüllt, auch ψ erfüllt.

Ein Beispiel

Wir betrachten

$$\phi := \big((X \vee Y) \wedge (\neg X \vee Y)\big) \text{ und } \psi := \big(Y \vee (\neg X \wedge \neg Y)\big).$$

- Wir stellen die Wahrheitstafel für ϕ und ψ auf:

X	Y	$(X \vee Y)$	$(\neg X \vee Y)$	$\neg X \wedge \neg Y$	ϕ	ψ
0	0	0	1	1	0	1
0	1	1	1	0	1	1
1	0	1	0	0	0	0
1	1	1	1	0	1	1

- In jeder Zeile, in der eine 1 in der Spalte von „ϕ“ steht, steht auch in der Spalte von „ψ“ eine 1.
  - Somit gilt $\phi \models \psi$.
- Aber in Zeile 1, also für die Belegung $\mathcal{B}$ mit $\mathcal{B}(X) = 0$ und $\mathcal{B}(Y) = 0$, steht in der Spalte von “ψ“ eine 1 und in der Spalte von “ϕ“ eine 0.
  - Es gilt also $[\![\psi]\!]^{\mathcal{B}} = 1$ und $[\![\phi]\!]^{\mathcal{B}} = 0$.
  - Daher gilt $\psi \not\models \phi$.

Wie passt das alles zusammen?

Seien ϕ und ψ beliebige aussagenlogische Formeln. Dann gilt:

(a) $\mathbf{1} \models \phi \iff \phi$ ist allgemeingültig,

(b) $\phi \models \mathbf{0} \iff \phi$ ist unerfüllbar,

(c) $\phi \models \psi \iff (\phi \to \psi)$ ist allgemeingültig und

(d) $\phi \models \psi \iff (\phi \wedge \neg\psi)$ ist unerfüllbar.

(Semantische) Äquivalenz

DEFINITION

1. Aussagenlogische Formeln ϕ und ψ heißen (semantisch) **äquivalent**, kurz:

$$\phi \equiv \psi,$$

wenn $\phi \models \psi$ und $\psi \models \phi$ gilt, oder in anderen Worten, wenn für alle zu ϕ und ψ passenden Belegungen $\mathcal{B}$ gilt:

$$[\![\phi]\!]^{\mathcal{B}} = [\![\psi]\!]^{\mathcal{B}}.$$

2. Zwei Mengen Φ und Ψ aussagenlogischer Formeln heißen (semantisch) **äquivalent**, kurz:

$$\Phi \equiv \Psi,$$

wenn $\Phi \models \psi$ für alle Formeln $\psi \in \Psi$ und $\Psi \models \phi$ für alle Formeln $\phi \in \Phi$ gilt.

ϕ und ψ sind genau dann äquivalent, wenn in ihrer gemeinsamen Wahrheitstafel die Spalten von ϕ und ψ übereinstimmen.

Ein Beispiel

Sind

$$\phi := \big(X \wedge (X \vee Y)\big) \text{ und } \psi := X$$

äquivalent?

- Wir stellen die Wahrheitstafel auf:

X	Y	$(X \vee Y)$	ϕ	ψ
0	0	0	0	0
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

- Die Spalte von „ϕ“ stimmt überein mit der Spalte von „ψ“: Es folgt

$$\phi \equiv \psi.$$

Wie passt das alles zusammen?

Seien ϕ und ψ aussagenlogische Formeln. Dann gilt:

(a) $\phi \equiv \psi \iff (\phi \leftrightarrow \psi)$ ist allgemeingültig $\iff \phi \models \psi$ und $\psi \models \phi$.

(b) ϕ ist allgemeingültig $\iff \phi \equiv \mathbf{1} \iff \mathbf{1} \models \phi$.

(c) ϕ ist unerfüllbar $\iff \phi \equiv \mathbf{0} \iff \phi \models \mathbf{0}$.

(d) ϕ ist erfüllbar $\iff \phi \not\equiv \mathbf{0}$, d.h. „$\phi \equiv \mathbf{0}$“ gilt nicht $\iff \phi \not\models \mathbf{0}$.

(e) ϕ ist falsifizierbar $\iff \phi \not\equiv \mathbf{1} \iff \mathbf{1} \not\models \phi$.

SymPy: (Semantische) Folgerung/Äquivalenz

Seien ϕ, ψ aussagenlogische Formeln und
seien `phi, psi` SymPy-Formeln, die zu ϕ bzw. ψ äquivalent sind.

Wir arbeiten in SymPy Live (Benutze das Setting "Submit with Shift-Enter"):

(a) Wir definieren die Funktion `implication()`

```
>>> def implication( phi, psi ):
....         return satisfiable(Not(Implies( phi, psi ))) == False
```

$\phi \models \psi$ gilt genau dann, wenn `implication( phi, psi )` den Wert `True` ausgibt.

(b) Wir definieren die Funktion `equivalence()`

```
>>> def equivalence( phi, psi ):
....         return implication(phi,psi) & implication(psi,phi)
```

$\phi \equiv \psi$ gilt genau dann, wenn `equivalence( phi, psi )` den Wert `True` ausgibt.

Fundamentale Äquivalenzen (1/2)

Seien ϕ, ψ und χ aussagenlogische Formeln. Dann gilt:

(a) **Idempotenz:**
- $(\phi \wedge \phi) \equiv \phi$
- $(\phi \vee \phi) \equiv \phi$

(b) **Kommutativität:** Für $\circ \in \{\vee, \wedge, \leftrightarrow, \oplus\}$ gilt
- $(\phi \circ \psi) \equiv (\psi \circ \phi)$

(c) **Assoziativität:** Für $\circ \in \{\vee, \wedge, \leftrightarrow, \oplus\}$ gilt
- $((\phi \circ \psi) \circ \chi) \equiv (\phi \circ (\psi \circ \chi))$

(d) **Absorption:**
- $(\phi \wedge (\phi \vee \psi)) \equiv \phi$
- $(\phi \vee (\phi \wedge \psi)) \equiv \phi$

(e) **Distributivität:**
- $(\phi \wedge (\psi \vee \chi)) \equiv ((\phi \wedge \psi) \vee (\phi \wedge \chi))$
- $(\phi \vee (\psi \wedge \chi)) \equiv ((\phi \vee \psi) \wedge (\phi \vee \chi))$

Fundamentale Äquivalenzen (2/2)

(f) **Doppelte Negation:**
- $\neg\neg\phi \equiv \phi$

(g) **De Morgansche Regeln:**
- $\neg(\phi \wedge \psi) \equiv (\neg\phi \vee \neg\psi)$
- $\neg(\phi \vee \psi) \equiv (\neg\phi \wedge \neg\psi)$

(h) **Tertium non Datur:**
- $(\phi \wedge \neg\phi) \equiv \mathbf{0}$
- $(\phi \vee \neg\phi) \equiv \mathbf{1}$

(i) **True/False 1**
- $(\phi \wedge \mathbf{1}) \equiv \phi$
- $(\phi \wedge \mathbf{0}) \equiv \mathbf{0}$
- $(\phi \vee \mathbf{1}) \equiv \mathbf{1}$
- $(\phi \vee \mathbf{0}) \equiv \phi$

(j) **True/False 2**
- $\mathbf{1} \equiv \neg\mathbf{0}$
- $\mathbf{0} \equiv \neg\mathbf{1}$

(k) **Elimination der Implikation:**
- $(\phi \rightarrow \psi) \equiv (\neg\phi \vee \psi) \equiv \neg(\phi \wedge \neg\psi)$

(l) **Elimination von Äquivalenz und XOR:**
- $\neg(\phi \oplus \psi) \equiv (\phi \leftrightarrow \psi) \equiv ((\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi))$

Mengengleichheiten $\Longleftrightarrow$ semantische Äquivalenzen

Für eine Mengengleichheit $X = Y$ wie etwa das Distributivgesetz

$$X := M_1 \cap (M_2 \cup M_3) = (M_1 \cap M_2) \cup (M_1 \cap M_3) =: Y$$

bestimme eine „zugehörige“ semantische Äquivalenz $\phi_X \equiv \phi_Y$.

1. Ersetze M_i jeweils durch die aussagenlogische Variable $V_i \Longrightarrow$

$$V_1 \cap (V_2 \cup V_3) = (V_1 \cap V_2) \cup (V_1 \cap V_3)$$

2. Ersetze $\cap, \cup, \oplus$ durch die Junktoren $\wedge, \vee, \oplus \Longrightarrow$

$$V_1 \wedge (V_2 \vee V_3) = (V_1 \wedge V_2) \vee (V_1 \wedge V_3)$$

   Übrigens, wie ist die Mengendifferenz $U \setminus W$ zu behandeln?

3. Ersetze „$=$“ durch „$\equiv$“ $\Longrightarrow$

$$\phi_X := V_1 \wedge (V_2 \vee V_3) \stackrel{!}{\equiv} (V_1 \wedge V_2) \vee (V_1 \wedge V_3) := \phi_Y$$

Ohne Beweis: $X = Y$ gilt für **alle** Mengen $M_1, \ldots, M_n \Longleftrightarrow \phi_X \equiv \phi_Y$

Zurück zum „Geburtstagsproblem“

Sei ϕ die Formel aus dem Geburtagsproblem. Die Frage

> *„Wie viele (und welche) Freunde werden bestenfalls zu Fred's Party kommen?“*

können wir lösen, indem wir

1. die Wahrheitstafel für ϕ ermitteln,
2. alle Zeilen heraussuchen,
   in denen in der mit „ϕ“ beschrifteten Spalte der Wert 1 steht und
3. aus diesen Zeilen all jene heraussuchen, bei denen in den mit A, B, C, D, E beschrifteten Spalten möglichst viele Einsen stehen.

   > Jede dieser Zeilen repräsentiert dann eine größtmögliche Konstellation von gleichzeitig erscheinenden Freunden.

A	B	C	D	E	$E \to (C \wedge D)$	$C \to A$	$(B \wedge E) \to \neg D$	$A \to (B \vee C)$	$(B \wedge A) \to \neg E$	φ
0	0	0	0	0	1	1	1	1	1	1
0	0	0	0	1	0	1	1	1	1	0
0	0	0	1	0	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	0
0	0	1	0	0	1	0	1	1	1	0
0	0	1	0	1	0	0	1	1	1	0
0	0	1	1	0	1	0	1	1	1	0
0	0	1	1	1	1	0	1	1	1	0
0	1	0	0	0	1	1	1	1	1	1
0	1	0	0	1	0	1	1	1	1	0
0	1	0	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	0	1	1	0
0	1	1	0	0	1	0	1	1	1	0
0	1	1	0	1	0	0	1	1	1	0
0	1	1	1	0	1	0	1	1	1	0
0	1	1	1	1	1	0	0	1	1	0
1	0	0	0	0	1	1	1	0	1	0
1	0	0	0	1	0	1	1	0	1	0
1	0	0	1	0	1	1	1	0	1	0
1	0	0	1	1	0	1	1	0	1	0
1	0	1	0	0	1	1	1	1	1	1
1	0	1	0	1	0	1	1	1	1	0
1	0	1	1	0	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1
1	1	0	0	0	1	1	1	1	1	1
1	1	0	0	1	0	1	1	1	0	0
1	1	0	1	0	1	1	1	1	1	1
1	1	0	1	1	0	1	0	1	0	0
1	1	1	0	0	1	1	1	1	1	1
1	1	1	0	1	0	1	1	1	0	0
1	1	1	1	0	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	0	0

Das Geburtstagsproblem: Die Lösung

Es gibt keine Zeile mit genau 5 Einsen, aber genau zwei Zeilen mit insgesamt 4 Einsen in den mit A bis E beschrifteten Spalten:

Es handelt sich um die beiden Belegungen $\mathcal{B}_1$ und $\mathcal{B}_2$ mit

$$\mathcal{B}_1(A) = \mathcal{B}_1(C) = \mathcal{B}_1(D) = \mathcal{B}_1(E) = 1 \quad \text{und} \quad \mathcal{B}_1(B) = 0$$

und

$$\mathcal{B}_2(A) = \mathcal{B}_2(B) = \mathcal{B}_2(C) = \mathcal{B}_2(D) = 1 \quad \text{und} \quad \mathcal{B}_2(E) = 0.$$

Bestenfalls werden 4 der 5 Freunde kommen, und dafür gibt es zwei Möglichkeiten, nämlich

(1) dass alle außer Bernd kommen, und

(2) dass alle außer Eva kommen.

Müssen wir die Wahrheitstafel vollständig bestimmen?

Für das Geburtstagsproblem ist es völlig ausreichend, wenn wir

(a) uns zuerst überzeugt hätten, dass alle fünf Freunde nicht zugleich zur Party kommen werden,

(b) und dann die **fünf** Belegungen untersucht hätten, in denen genau ein Freund nicht kommt.

(c) Warum stur die letzten sechs Spalten berechnen?

- Hat eine der Spalten 6-10 den Wert „falsch“ erhalten, ist ϕ falsch und der Wert der verbleibenden Spalten ist uninteressant.

Bitte schön:
Immer mit Köpfchen, denn Wahrheitstafeln werden sehr schnell sehr groß!

Wie groß sind denn Wahrheitstafeln?

Die Größe von Wahrheitstafeln (1/2)

Sei ϕ eine aussagenlogische Formel. Wie viele Belegungen hat ϕ, d.h. wieviele Funktionen $\mathcal{B} : \mathrm{Var}(\phi) \to \{0,1\}$ gibt es? Genau

$$|\mathrm{Abb}(\mathrm{Var}(\phi), \{0,1\})| = |\{0,1\}|^{|\mathrm{Var}(\phi)|} = 2^{|\mathrm{Var}(\phi)|}$$

viele.

Sei

$$n := |\mathrm{Var}(\phi)|$$

die Anzahl der in ϕ vorkommenden Variablen. Dann gibt es

$$\mathbf{2^n}$$

verschiedene zu ϕ passende Belegungen $\mathcal{B} : \mathrm{Var}(\phi) \to \{0,1\}$.

Die Größe von Wahrheitstafeln (2/2)

Die Wahrheitstafel einer Formel mit n Variablen hat genau 2^n Zeilen.

Ist das gut oder schlecht?

n (Anzahl Variablen)	2^n	(Anzahl Zeilen der Wahrheitstafel)
10	$2^{10} =$	$1.024 \approx 10^3$
20	$2^{20} =$	$1.048.576 \approx 10^6$
30	$2^{30} =$	$1.073.741.824 \approx 10^9$
40	$2^{40} =$	$1.099.511.627.776 \approx 10^{12}$
50	$2^{50} =$	$1.125.899.906.842.624 \approx 10^{15}$
60	$2^{60} =$	$1.152.921.504.606.846.976 \approx 10^{18}$

Zum Vergleich: Das Alter des Universums wird auf $13,7$ Milliarden Jahre, das sind ungefähr 10^{18} Sekunden, geschätzt.

Wir bauen eine disjunktive Normalform
aus einer Wahrheitstafel

Literale, Konjunktionsterme und disjunktive Normalformen

(a) Ein **Literal** ℓ ist eine Formel der Form $\ell = X$ oder $\ell = \neg X$ mit $X \in \mathrm{AVAR}$, d.h. X ist eine Variable.

- Das Literal X wird auch **positives Literal** genannt,
- das Literal $\neg X$ heißt **negatives Literal**.

(b) Eine Konjunktion von (positiven oder negativen) Literalen heißt ein **Konjunktionsterm**.

(c) Eine Disjunktion

$$\psi = \bigvee_{i=1}^{k} \left(\bigwedge_{j=1}^{m_i} \ell_{i,j} \right)$$

von Konjunktionstermen $\bigwedge_{j=1}^{m_i} \ell_{i,j}$ ist in

disjunktiver Normalform (DNF),

wenn $k, m_1, \ldots, m_k \in \mathbb{N}_{>0}$ und $\ell_{i,j}$ für jedes $i \in \{1, \ldots, k\}, j \in \{1, \ldots, m_i\}$ ein Literal ist.

- Gilt $\phi \equiv \psi$, dann heißt ψ eine **DNF für** ϕ, bzw. ϕ **besitzt die DNF** ψ.

1. Gibt es zu jeder Wahrheitstafel eine äquivalente DNF?
2. Wie konstruiert man DNFs?

Wahrheitstafel $\Rightarrow$ DNF (1/3)

Die Wahrheitstafel T sei gegeben: Baue eine DNF ψ mit T als Wahrheitstafel.

1. Sei z eine Zeile der Wahrheitstafel T
   - z heißt eine **1-Zeile**, wenn in der Spalte des Wahrheitswerts eine „1“ steht.
   - Steht in der Spalte des Wahrheitswerts eine „0“, nennen wir z eine **0-Zeile**.
2. Die Menge aller Variablen der Wahrheitstafel T bezeichnen wir mit $\mathrm{Var}(T)$.
3. Für jede 1-Zeile z mit Belegung $\mathcal{B}_z : \mathrm{Var}(T) \to \{0,1\}$ definieren wir die **Vollkonjunktion Konj_z** von **z** als den Konjunktionsterm
   - in dem **jede** Variable $V \in \mathrm{Var}(T)$ als Literal V oder $\neg V$ vorkommt und
   - der von $\mathcal{B}_z$ erfüllt, aber von den Belegungen zu anderen Zeilen falsifiziert wird.

Die Vollkonjunktion der 1-Zeile z hat also die Form

$$\mathrm{Konj}_z = \Big(\bigwedge_{V \in \mathrm{Var}(T) : \mathcal{B}_z(V) = 1} V \Big) \wedge \Big(\bigwedge_{V \in \mathrm{Var}(T) : \mathcal{B}_z(V) = 0} \neg V \Big).$$

Wahrheitstafel $\Rightarrow$ DNF (2/3)

Die Wahrheitstafel T sei gegeben: Baue eine DNF ψ mit T als Wahrheitstafel.

1. Wie gerade gesehen hat die Vollkonjunktion der 1-Zeile z die Form

$$\mathrm{Konj}_z = \Big(\bigwedge_{V \in \mathrm{Var}(T):\mathcal{B}_z(V)=1} V \Big) \wedge \Big(\bigwedge_{V \in \mathrm{Var}(T):\mathcal{B}_z(V)=0} \neg V \Big).$$

2. Die Disjunktion

$$\psi := \bigvee_{z \text{ ist eine 1-Zeile}} \mathrm{Konj}_z,$$

also die „Ver*oder*ung“ aller Vollkonjunktionen zu 1-Zeilen der Wahrheitstafel, heißt

kanonische DNF der Wahrheitstafel.

DNFs: Ein erstes Beispiel

Betrachte die Wahrheitstafel T:

X	Y	Z	ϕ
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Es gibt genau drei 1-Zeilen (für die in der „Spalte von ϕ“ eine 1 steht), nämlich

X	Y	Z	ϕ	zur Belegung der Zeile gehörende Vollkonjunktion:
0	0	0	1	$\neg X \wedge \neg Y \wedge \neg Z$
1	0	0	1	$X \wedge \neg Y \wedge \neg Z$
1	0	1	1	$X \wedge \neg Y \wedge Z$

Wir erhalten die zur Wahrheitstafel T passende kanonische DNF

$$\psi \quad := \quad (\neg X \wedge \neg Y \wedge \neg Z) \ \vee \ (X \wedge \neg Y \wedge \neg Z) \ \vee \ (X \wedge \neg Y \wedge Z).$$

Wahrheitstafel $\Rightarrow$ DNF (3/3)

Die Wahrheitstafel T sei gegeben: Baue eine DNF ψ mit T als Wahrheitstafel.

Die **kanonische DNF** von T ist

$$\psi := \bigvee_{z \text{ ist eine 1-Zeile}} \underbrace{\left(\left(\bigwedge_{V \in \mathrm{Var}(T):\mathcal{B}_z(V)=1} V \right) \wedge \left(\bigwedge_{V \in \mathrm{Var}(T):\mathcal{B}_z(V)=0} \neg V \right) \right)}_{=\mathrm{Konj}_z}.$$

Es gilt $[\![\mathrm{Konj}_u]\!]^{\mathcal{B}_v} = 1$ genau dann, wenn $u = v$.

(a) Ist u eine 1-Zeile von T, dann ist Konj_u **ein** Konjunktionsterm von $\psi \Longrightarrow$

$$[\![\psi]\!]^{\mathcal{B}_u} = [\![\bigvee_{z \text{ ist eine 1-Zeile}} Konj_z]\!]^{\mathcal{B}_u} = 1.$$

(b) Ist u eine 0-Zeile von T, dann ist Konj_u **kein** Konjunktionsterm von ψ und $[\![\mathrm{Konj}_z]\!]^{\mathcal{B}_u} = 0$ gilt für alle 1-Zeilen $z \Longrightarrow$

$$[\![\psi]\!]^{\mathcal{B}_u} = 0.$$

ψ besitzt T als Wahrheitstafel.

Jede Wahrheitstafel besitzt eine DNF

(a) Zu jeder Wahrheitstafel T gibt es eine DNF ψ, so dass T die Wahrheitstafel von ψ ist.
   - Wir sagen, dass ψ eine DNF für T ist.

(b) Insbesondere besitzt jede aussagenlogische Formel ϕ eine DNF.
   - Wir sagen, dass ψ eine DNF für ϕ ist.

SymPy: Berechnung einer DNF

Sei ϕ eine aussagenlogische Formel und sei `phi` eine zu ϕ äquivalente SymPy-Formel.

Die SymPy-Funktion `to_dnf` berechnet eine DNF für ϕ mit dem Aufruf

```
>>> to_dnf( phi )
```

Für die Ausgabe einer möglichst kurzen DNF wähle den Aufruf

```
>>> to_dnf( phi , simplify = True)
```

Die konjunktive Normalform

Disjunktionsterme und KNFs

- Ein **Disjunktionsterm** (oder eine **Klausel**) ist eine Disjunktion von Literalen.
- Eine Konjunktion

$$\psi = \bigwedge_{i=1}^{k} \left(\bigvee_{j=1}^{m_i} \ell_{i,j} \right)$$

von Disjunktionstermen $\bigvee_{j=1}^{m_i} \ell_{i,j}$ ist in **konjunktiver Normalform (KNF)**, wenn $k, m_1, \ldots, m_k \in \mathbb{N}_{>0}$ und $\ell_{i,j}$ für jedes $i \in \{1, \ldots, k\}, j \in \{1, \ldots, m_i\}$ ein Literal ist.

  - Sei ϕ eine aussagenlogische Formel. Ist

$$\phi \equiv \psi,$$

dann heißt ψ eine **KNF für** ϕ, bzw. wir sagen, dass ϕ die **KNF** ψ **besitzt**.

Wahrheitstafel $\Rightarrow$ KNF

Die Wahrheitstafel T sei gegeben: Baue eine KNF ψ mit T als Wahrheitstafel.

1. Invertiere die Wahrheitstafel: Ersetze in jeder Zeile den Wahrheitswert für ϕ durch $\neg\phi$.
2. Baue eine DNF

$$\psi = \bigvee_{i=1}^{k} \left(\bigwedge_{j=1}^{m_i} \ell_{i,j} \right)$$

   für die negierte Wahrheitstafel, also für $\neg\phi$.
3. Wende die **De Morgan Regeln** an:

$$\neg\psi = \neg \bigvee_{i=1}^{k} \left(\bigwedge_{j=1}^{m_i} \ell_{i,j} \right) \equiv \bigwedge_{i=1}^{k} \left(\bigvee_{j=1}^{m_i} \neg\ell_{i,j} \right) =: \psi'$$

   ist eine **KNF** für $\neg\neg\phi \equiv \phi$.
   - Ist ψ eine kanonische DNF für $\neg\phi$, dann ist $\neg\psi$ eine **kanonische KNF** für ϕ.

KNFs: Beispiele

Wir betrachten die Wahrheitstafel T:

X	Y	Z	ϕ
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Wie sieht die kanonische KNF in diesem Beispiel aus? Siehe Tafel

DNFs und KNFs

Für jede aussagenlogische Formel ϕ gibt es eine Formel ψ_D in DNF und eine Formel ψ_K in KNF, so dass

$$\phi \equiv \psi_D \text{ und } \phi \equiv \psi_K.$$

Jede Formel ist äquivalent zu einer Formel in DNF und zu einer Formel in KNF.

Normalformen spielen in vielen Anwendungsgebieten eine wichtige Rolle.

- Beispielsweise geht man in der Schaltungstechnik (Hardware-Entwurf) oft von DNF- oder KNF-Formeln aus,
- während bei der Wissensrepräsentation häufig KNF-Formeln auftreten, da hier viele, aber **einfach strukturierte** Aussagen „verundet“ werden.

KNFs: Beispiele

(a) **DNFs** für

$$\phi_n = \bigwedge_{i=1}^{n} \Big(X_i \leftrightarrow Y_i \Big)$$

benötigen **viele** Konjunktionsterme. Aber ϕ_n hat doch eine **„kleine“ KNF :-)))**

(b) Leider hat

$$\psi_n = \Big(X_1 \oplus \cdots \oplus X_n \Big)$$

nur DNFs oder KNFs mit mindestens 2^{n-1} Termen.

Wir spielen Sudoku

Können wir Sudokus erfolgreich mit Hilfe von KNFs vervollständigen?

						9		
7	9			6				
		5		3			1	4
	4				6			7
		3	2	1		5		
	7						8	
					1		9	2
			5			7		
	1			9	4		5	

Sudokus mit KNFs modellieren (1/3)

1. Benutze die Variablen $Z_{i,j}^k$ für $1 \leqslant i, j, k \leqslant 9$.
   - $Z_{i,j}^k = 1$ soll bedeuten, dass in Zeile i und Spalte j die Zahl k steht.
2. Für jede **Zeile** i und jede **Ziffer** k fordern wir
   - im Disjunktionsterm $\mathbf{Mindestens}_i^k$, dass die Ziffer k mindestens einmal erscheint

$$\mathbf{Mindestens}_i^k := \bigvee_{j=1}^{9} Z_{i,j}^k.$$

   - Die Ziffer k soll in Zeile i nicht zweimal vergeben werden $\Longrightarrow$ für alle Spalten $j \neq j^*$ mit $1 \leqslant j \leqslant j^* \leqslant 9$ und alle Ziffern k soll gelten

$$Z_{i,j}^k \to \neg Z_{i,j^*}^k.$$

   - Beachte die Äquivalenz $(Z_{i,j}^k \to \neg Z_{i,j^*}^k) \equiv (\neg Z_{i,j}^k \vee \neg Z_{i,j^*}^k)$. Wie drückt man aus, dass die Ziffer k genau einmal in Zeile i erscheint? Definiere die KNF

$$\mathbf{Zeile}_i^k := \mathbf{Mindestens}_i^k \wedge \bigwedge_{j=1}^{8} \bigwedge_{j^*=j+1}^{9} (\neg Z_{i,j}^k \vee \neg Z_{i,j^*}^k).$$

3. Wie drückt man aus, dass **jede Ziffer** genau einmal in **jeder Zeile** vorkommt? Definiere die KNF

$$\textbf{Zeilen} := \bigwedge_{i=1}^{9} \bigwedge_{k=1}^{9} \text{Zeile}_i^k.$$

4. Auf ähnliche Art definieren wir KNFs

Spalten und **Teilmatrizen**

für die Spalten und 3×3 Teilmatrizen, um diesmal auszudrücken, dass jede Ziffer in jeder Spalte, bzw. in jeder Teilmatrix genau einmal vorkommt.

5. Keine Zelle (i, j) darf zwei oder mehr Ziffern erhalten. Fordere

$$Z_{i,j}^k \to \neg Z_{i,j}^{k^*}$$

für alle $1 \leqslant i, j \leqslant 9$ und alle verschiedenen Ziffern k und k^*. Wie drücken wir aus, dass in **jeder** Zelle höchstens eine Ziffer stehen darf? Durch die KNF

$$\textbf{Einfach} := \bigwedge_{i=1}^{9} \bigwedge_{j=1}^{9} \bigwedge_{k=1}^{8} \bigwedge_{k^*=k+1}^{9} (\neg Z_{i,j}^k \vee \neg Z_{i,j}^{k^*}).$$

Sudokus mit KNFs modellieren (3/3)

6. Ist die Zelle in Zeile i und Spalte j mit der Ziffer k bereits gesetzt, dann füge den Disjunktionsterm

   $$Z_{i,j}^{k}$$

   hinzu und definiere die KNF

   Schon_da

   als Konjunktion dieser Disjunktionsterme für alle bereits gesetzten Zellen.

In der KNF

$$\textbf{Sudoku} := \textbf{Zeilen} \wedge \textbf{Spalten} \wedge \textbf{Teilmatrizen} \wedge \textbf{Einfach} \wedge \textbf{Schon\_da}$$

sind alle Sudoku-Regeln umgesetzt:

Genau die erfüllenden Belegungen lösen das Sudoku-Rätsel. (Warum haben wir nicht fordern müssen, dass jede Zelle mindestens eine Ziffer erhält?)

SymPy: Berechnung einer KNF

Sei ϕ eine aussagenlogische Formel und sei `phi` eine zu ϕ äquivalente SymPy-Formel.

Die SymPy-Funktion `to_cnf` berechnet eine KNF für ϕ mit dem Aufruf

```
>>> to_cnf( phi )
```

Für die Ausgabe einer möglichst kurzen KNF wähle den Aufruf

```
>>> to_cnf( phi , simplify = True)
```

Das KNF-Erfüllbarkeitsproblem (KNF-SAT)

Eingabe: Eine aussagenlogische Formel ϕ in KNF.

Frage: Ist ϕ erfüllbar?

Wie schwierig ist das Erfüllbarkeitsproblem? (1/2)

- Natürlich kann man das Erfüllbarkeitsproblem mit einer Wahrheitstafel lösen.
  - Teste, ob es in der mit „ϕ“ beschrifteten Spalte mindestens eine 1 gibt.
  - Aber die Wahrheitstafel hat **exponentiell** viele Zeilen, nämlich 2^n Zeilen, wenn ϕ von n Variablen abhängt.
- In der „**Theoretischen Informatik 1**“ (3. Semester) wird gezeigt:

Satz von Cook

KNF-SAT ist NP-vollständig.

Wie schwierig ist das Erfüllbarkeitsproblem? (2/2)

- Eine präzise Definition des Begriffs „NP-vollständig“ wird in der **Theoretischen Informatik 1** gegeben.
  - Grob gesagt bedeutet die NP-Vollständigkeit von KNF-SAT, dass es vermutlich kein **effizientes** Verfahren gibt, das KNF-SAT für **alle** Eingabeformeln löst. **:-((**
    - ⋆ Ein Verfahren wird effizient gennant, wenn seine Laufzeit nur moderat mit der Länge der Formel wächst.
    - ⋆ Genauer: Die Laufzeit darf höchstens **polynomiell** in der Eingabelänge wachsen!
- Einige Heuristiken, die so genannten **SAT-Solver**, lösen KNF-SAT trotzdem für **viele** Eingabe-Formeln **erstaunlich effizient**.
  - Ich muss hoffen, dass ich ein Verfahren finde, dass **meine** Formel schnell genug löst.
  - Aber es gibt wohl kein „einigermaßen“ schnelles Verfahren für **alle** Formeln!

Wie arbeiten denn diese SAT-Solver?

SAT-Solver: Das Beweisverfahren der Resolution

Das Beweisverfahren der Resolution (1/2)

Sei Φ eine Menge von Disjunktionstermen.

(a) Stelle einen **Disjunktionsterm**

$$\gamma = \ell_1 \vee \cdots \vee \ell_m$$

als **Menge**

$$\{\ell_1, \ldots, \ell_m\}$$

seiner Literale dar.

(b) **Die Resolutionsregel:** Sei X eine Variable, α, β seien Disjunktionsterme. Aus den Disjunktionstermen $\alpha \cup \{X\}$ und $\beta \cup \{\neg X\}$ darf der „Resolutionsterm“ $\alpha \cup \beta$ abgeleitet werden. Man schreibt auch kurz

$$\frac{\alpha \cup \{X\},\ \beta \cup \{\neg X\}}{\alpha \cup \beta}$$

Das Beweisverfahren der Resolution (2/2)

(c) Ein **Resolutionsbeweis**

$$\Phi \vdash_{\mathfrak{R}} \chi$$

eines Disjunktionsterms χ aus einer Menge Φ von Disjunktionstermen ist ein Tupel

$$(\chi_1, \ldots, \chi_i, \ldots, \chi_n)$$

von Disjunktionstermen. Es muss gelten:

1. χ_n ist die Zielformel, d.h. es gilt $\chi_n = \chi$.
2. Für jedes i mit $1 \leqslant i \leqslant n$ ist
   - χ_i eine Formel aus Φ oder
   - es gibt $1 \leqslant i_1 < i_2 < i$ und χ_i folgt aus χ_{i_1}, χ_{i_2} mit der Resolutionsregel.

Resolution: Ein erstes Beispiel

Zeige die „Transitivität“ der Implikation, d.h. **zeige**

$$\Phi \vdash_{\mathfrak{R}} \chi$$

mit

$$\Phi := \{\{\neg X, Y\}, \{\neg Y, Z\}\} \text{ und } \chi := \{\neg X, Z\}.$$

1. Die Formel $\chi_1 := \{\neg X, Y\}$ gehört zu Φ und ist deshalb in $\mathfrak{R}$ ableitbar.
2. Gleiches gilt für die Formel $\chi_2 := \{\neg Y, Z\}$ und deshalb ist auch χ_2 in $\mathfrak{R}$ ableitbar.
3. Jetzt erhalten wir $\chi_3 := \{\neg X, Z\} = \chi$ nach Anwendung der Resolutionsregel

$$\frac{\{\neg X, Y\}, \{\neg Y, Z\}}{\{\neg X, Z\}}$$

auf χ_1 und χ_2.

KNF-SAT mit Resolutionsbeweisen lösen?!

Ist die KNF $D_1 \wedge D_2 \wedge \cdots \wedge D_m$ erfüllbar?

Sei $\Phi := \{ D_1, \ldots, D_m \}$.

:-)) Wenn es einen Resolutionsbeweis $\Phi \vdash_{\mathfrak{R}} \epsilon$ des leeren Disjunktionsterms ϵ gibt, dann ist Φ **unerfüllbar**!

:-)) Im Skript wird gezeigt: Sei Φ eine Menge von Disjunktionstermen. Dann gilt

$$\left(\bigwedge_{D \in \Phi} D \right) \text{ ist unerfüllbar} \iff \Phi \vdash_{\mathfrak{R}} \epsilon.$$

.

ϕ ist also **genau dann** unerfüllbar, wenn der leere Disjunktionsterm aus Φ mit einem Resolutionsbeweis hergeleitet werden kann.

:-((Leider, leider gibt es aber Formelmengen Φ deren Unerfüllbarkeit nur mit **exponentiell** vielen Resolutionsschritten nachgewiesen werden kann!

- In diesen Fällen ist nicht nur das Finden eines „Resolutionsbeweises“ kompliziert,
- sondern mgl. gibt es nur sehr, sehr lange Beweise.

Resolution: Ein zweites Beispiel

1. Die Kunden der Bahn sind nicht **z**ufrieden, wenn
   - sich die **P**reise erhöhen: $P \to \neg Z$ folgt,
   - oder sich die **F**ahrzeiten verlängern, d.h. $F \to \neg Z$ ist die Konsequenz.
2. Wenn der Frankfurter Kopfbahnhof nicht in einen Durchgangsbahnhof umge**b**aut wird, verlängern sich die Fahrzeiten, also gilt $\neg B \to F$.
3. Der Bahnhof kann nur dann umgebaut werden, wenn die Fahrpreise erhöht werden, d.h. $B \to P$ folgt.

Die Bahn kann es niemandem recht machen, denn die Formelmenge

$$\Phi := \Big\{ \{\neg P, \neg Z\}, \{\neg F, \neg Z\}, \{B, F\}, \{\neg B, P\}, \{Z\} \Big\}$$

ist unerfüllbar.

Wie sieht ein Resolutionsbeweis $\Phi \vdash_{\mathfrak{R}} \epsilon$ aus?

DPLL-Verfahren: Moderne Varianten von Resolutionsbeweisen

1. Wenn $\Phi = \emptyset$, dann halte mit der Antwort „Φ ist erfüllbar“.
2. Wenn $\epsilon \in \Phi$, dann halte mit der Antwort „Φ ist unerfüllbar“.
3. „**Unit-Resolution**“: Wenn $X \in \Phi$ für eine Variable X, dann setze $X = 1$. D.h. entferne alle Disjunktionsterme aus Φ, die X enthalten und entferne jedes Auftreten von $\neg X$ in einem Term von Φ. (Behandle den Fall $\neg X \in \Phi$ analog.)
4. „**Pure Literal Rule**“: Wenn eine Variable X nur positiv in Termen von Φ vorkommt, dann entferne alle Terme aus Φ, in denen X vorkommt. (Die Setzung $X = 1$ kann nicht falsch sein.) Analog, wenn X nur negiert vorkommt.
5. „**Choose Literal**“. Eine Variable X wird ausgewählt. Führe „**Backtracking**“ mit X aus:
   - Rufe das Verfahren rekursiv für $\Phi' := \Phi \cup \{X\}$ auf.
   - Bei Antwort„unerfüllbar“, rufe Verfahren rekursiv für $\Phi' := \Phi \cup \{\neg X\}$ auf.

Boolesche Funktionen und Formeln

Boolesche Funktionen

Eine Funktion $f : \{0,1\}^n \to \{0,1\}$ heißt eine **boolesche Funktion**.

Beispiele:

(a) Die boolesche Funktion

$$\mathbf{Und}_n : \{0,1\}^n \to \{0,1\}$$

nimmt genau dann den Wert 1 für Eingabe x an, wenn $x = 1^n$.

(b) Die **Paritätsfunktion** $p_n : \{0,1\}^n \to \{0,1\}$ nimmt genau dann den Wert 1 für Eingabe x an, wenn x ungerade viele Einsen besitzt.

- Das „Paritätsbit“ $p_n(x_1, \ldots, x_n)$ ändert sich, wenn genau ein Bit „geflippt“ wird und wird deshalb zur Fehlererkennung eingesetzt.

(c) **Addition**$_i : \{0,1\}^{2n} \to \{0,1\}$ ist das $i$te Bit der Summe von zwei n-Bit Zahlen.

Boolesche Funktionen, Wahrheitstafeln und Formeln (1/2)

(a) In einer Wahrheitstafel

- werden die aussagenlogischen Variablen benannt (z.B. seien dies $X_1, \ldots, X_n$) und ihre Reihenfolge wird festgelegt.
- Dann wird für jede Belegung $\mathcal{B} : \{X_1, \ldots, X_n\} \to \{0, 1\}$ eine eigene Zeile angelegt und der Wahrheitswert

$$f([\![X_1]\!]^{\mathcal{B}}, \ldots, [\![X_n]\!]^{\mathcal{B}})$$

für $\mathcal{B}$ eingetragen. Nenne f **die boolesche Funktion der Wahrheitstafel**.

(b) Die Wahrheitstafel beschreibt f, tut allerdings ein klitzeklein wenig mehr:

- Sie benennt auch die aussagenlogischen Variablen und
- gibt ihnen eine Reihenfolge.

Die wesentlichen Eigenschaften einer Wahrheitstafel werden durch ihre boolesche Funktion ausgedrückt.

Boolesche Funktionen, Wahrheitstafeln und Formeln (2/2)

(c) Zu jeder Wahrheitstafel T gibt es eine DNF ψ, so dass ψ die Wahrheitstafel T besitzt. $\Longrightarrow$ Zu jeder Wahrheitstafel gibt es eine „äquivalente“ Formel.

(d) Stelle die Wahrheitstafel einer booleschen Funktion auf:
Auch zu jeder booleschen Funktion gibt es eine „äquivalente“ Formel, nämlich die zur Wahrheitstafel äquivalente Formel.

- Boolesche Funktionen, Wahrheitstafeln und Formeln sind unterschiedliche, aber äquivalente Beschreibungen
- Der Entwurf von Schaltungen für boolesche Funktionen ist ein wichtiges Ziel der technischen Informatik.

Haben boolesche Funktionen immer kleine DNFs oder kleine KNFs?

Definiere die Funktion **Gerade**$_n : \{0,1\}^n \to \{0,1\}$ mit

$$\text{Gerade}_n(x_1, \ldots, x_n) = 1 :\Longleftrightarrow (x_1, \ldots, x_n) \text{ besitzt gerade viele Einsen.}$$

(a) Jede DNF für Gerade$_n$ hat mindestens $\mathbf{2^{n-1}}$ Konjunktionsterme und

(b) jede KNF für Gerade$_n$ hat mindestens $\mathbf{2^{n-1}}$ Disjunktionsterme.

Es gibt also boolesche Funktionen, die weder „kleine“ DNFs noch „kleine“ KNFs besitzen!

Zusammenfassung: Aussagenlogik

(a) Belegungen einer aussagenlogischen Formel ϕ sind Funktionen

$$\mathcal{B} : \mathrm{Var}(\phi) \to \{0, 1\},$$

die jeder Variable von ϕ einen Wahrheitswert zuweisen.

- Wenn $|\mathrm{Var}(\phi)| = n$, dann hat ϕ genau $\mathbf{2^n}$ Belegungen.

(b) Wichtige Konzepte für aussagenlogische Formeln:

- Erfüllbarkeit und Falsifizierbarkeit, Allgemeingültigkeit und Unerfüllbarkeit,
- semantische Folgerung und Äquivalenz.

(c) Normalformen:

- DNFs: „Verodere“ alle Vollkonjunktionen zu den 1-Zeilen der Wahrheitstafel.
- KNFs: K ist eine KNF für $\phi \iff \neg K$ ist eine DNF für $\neg\phi$.
  - ⋆ Das Erfülbarkeitsproblem ist sehr schwer.
  - ⋆ Eine KNF ist genau dann unerfüllbar, wenn der leere Disjunktionsterm mit Hilfe der Resolution herleitbar ist. (DPLL-Verfahren sind „State-of-the-Art“.)

(d) Verschieden Sichtweisen:

- Jede aussagenlogische Formel ist mit einer Wahrheitstafel äquivalent und
- jede Wahrheitstafel ist mit einer booleschen Funktion äquivalent.

Alles verstanden?
Dann mal ran an das Einstein-Rätsel!

Das Einstein-Rätsel: Wem gehört der Fisch?

Pro Haus: Eine Farbe, ein(e) Bewohner(in) mit einem Getränk und einer Zigarettenmarke sowie ein Tier. Die Häuser stehen in Reihe.

1. Der Brite lebt im roten Haus.
2. Die Schwedin besitzt einen Hund.
3. Der Däne trinkt gern Tee.
4. Das grüne Haus steht (direkt) links neben dem weißen Haus.
5. Die Person, die im grünen Haus wohnt, trinkt Kaffee.
6. Die Person, die Pall Mall raucht, hält einen Vogel.
7. Die Person, die im mittleren Haus wohnt, trinkt Milch.
8. Die Person, die im gelben Haus wohnt, raucht Dunhill.
9. Die Norwegerin lebt im ersten Haus.
10. Die Person, die Marlboro raucht, wohnt neben der Person mit der Katze.
11. Die Person mit dem Pferd lebt neben der Person, die Dunhill raucht.
12. Die Person, die Winfield raucht, trinkt gern Bier.
13. Die Norwegerin wohnt neben dem blauen Haus.
14. Der Deutsche raucht Rothmanns.
15. Die Person, die Marlboro raucht, lebt neben der Person, die Wasser trinkt.