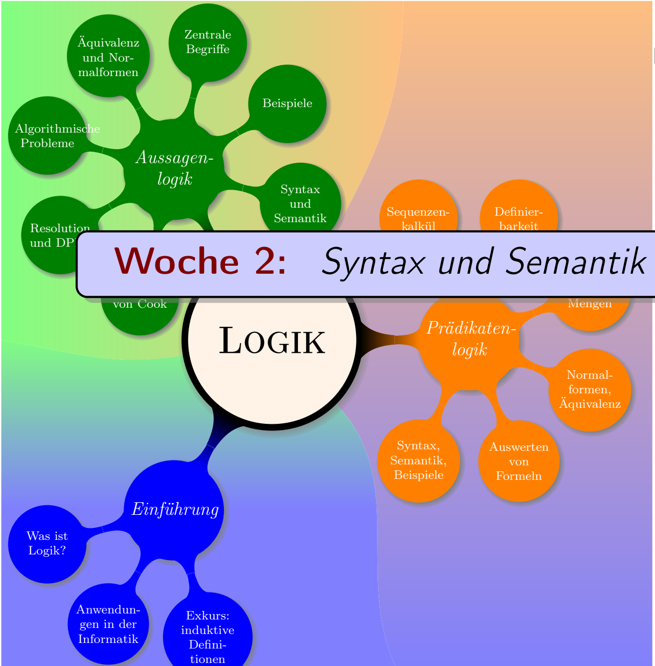
Logik im Wintersemester 2022/2023



Woche 2: *Syntax und Semantik der Aussagenlogik*

Nov.	17	18	19	20	21	22	23	*Einführung*
	24	25	26	27	28	29	30	*Aussagenlogik*
	31	1	2	3	4	5	6	*Normalformen*
	7	8	9	10	11	12	13	*Resolution*
Dez.	14	15	16	17	18	19	20	*Kompaktheit*
	21	22	23	24	25	26	27	*DPLL, Satz von Cook*
								Strukturen und FO
	5	6	7	8	9	10	11	*Prädikatenlogik*
	12	13	14	15	16	17	18	*Komplexität von FO*
Jan.	19	20	21	22	23	24	25	*Weihnachten*
	26	27	28	29	30	31	1	*Neujahr*
	2	3	4	5	6	7	8	*Normalformen*
	9	10	11	12	13	14	15	*Definierbarkeit*
	16	17	18	19	20	21	22	*EF-Spiele*
Feb.	23	24	25	26	27	28	29	*EF-Spiele*
	30	31	1	2	3	4	5	*Sequenzenkalkül AL*
	6	7	8	9	10	11	12	*Sequenzenkalkül FO*
	13	14	15	16	17	18	19	*Ausblick*

2.1 Organisatorisches

Wer wir sind

Fachgebietsleitung.

Stephan Kreutzer, `stephan.kreutzer@tu-berlin.de`

Wissenschaftliche Mitarbeiter.

Maximilian Gorsky, `m.gorsky@tu-berlin.de`

(Dario Cavallaro)

Tutor:innen.

Till,

Michelle,

Johannes,

Sebastian,

Elias

Vorlesung und Übungen

Termine.

Vorlesung	Do, 10-12	HE 101	Stephan Kreutzer
Großübung	Fr, 10-12 (ab 11.11.)	Zoom	Max Gorsky
Tutorien			

Wöchentliche Hausaufgaben.

Wir veröffentlichen jede Woche ein Hausaufgabenblatt.

Die Bearbeitung ist freiwillig.

Die Abgaben werden korrigiert und zurückgegeben.

Die Lösungen werden in der Großübung besprochen.

Portfolioprüfung

Portfolioprüfung. Anmeldung bis 23.11.2022

1. Hausarbeit	Abgabe: 7.12.2022	ISIS	10 PP
1. LK (MC-Test)	16.12.2022, 10:00 - 12:00	ISIS	30 PP
2. Hausarbeit	Abgabe: 9.02.2023	ISIS	20 PP
2. LK	04.03.2023, 15:30 - 18:30	Präsenz	40 PP

Es gilt der Notenschlüssel 1 der Fakultät IV.

Schriftliche Hausarbeiten.

Es gibt zwei schriftliche Hausarbeiten als Portfolioelemente.

Abgabe in Gruppen von 2 - 4 Personen.

Materialien

Folien.

Der Foliensatz für die gesamte Vorlesung wird auf ISIS veröffentlicht.

Nach jeder Vorlesung veröffentlichen wir die annotierten Folien auf der ISIS-Seite.

Skript. Es ausführliches Skript finden Sie auf der ISIS-Seite.

Videos.

Die Videos der letzten Jahre finden Sie auf der ISIS-Seite.

Hinweis. Wir erwarten nicht, dass Sie die Videos anschauen.

Inhaltlich sind die Videos mit der Vorlesung weitestgehend gleich.

Sonstiges.

Wir veröffentlichen verschiedenes weiteres Übungsmaterial während des Semesters.

Sonstiges

Lassen Sie sich nicht durch Kommentare vergangener Jahrgänge irritieren.

Bei Problemen, melden Sie sich bei uns.

Viel Erfolg

2.2 Einführung: Was ist Logik?

Logik?

Ursprünge der Logik. Das Wort *Logik* stammt aus dem Altgriechischen „Kunst des Denkens“ oder „Kunst des Argumentierens“.

Beispiel.

- Alle Vögel können fliegen.
- Tweety ist ein Vogel.
- Also kann Tweety fliegen.

Stimmt das Argument?

Wir können das Argument auf zwei Ebenen diskutieren: Inhaltlich.

Können alle Vögel fliegen? Wer oder was ist Tweety? Are birds real?

Logik?

Ursprünge der Logik. Das Wort *Logik* stammt aus dem Altgriechischen „Kunst des Denkens“ oder „Kunst des Argumentierens“.

Beispiel.

- Alle Vögel können fliegen.
- Tweety ist ein Vogel.
- Also kann Tweety fliegen.

Stimmt das Argument?

Wir können das Argument auf zwei Ebenen diskutieren: Abstrakt

In dem Beispiel wird eine Schlussfolgerung aus gemachten Annahmen gezogen.

Ist das überhaupt korrekt so?

Kann man aus den Annahmen die gemachte Behauptung wirklich folgern?

Logik?

Beispiel

Annahmen.

- Alle Vögel können fliegen.
- Tweety ist ein Vogel.

Schlussfolgerung.

- Also kann Tweety fliegen.

Beispiel

Annahmen.

- Alle Ersetzungschiffren sind anfällig für brute-force Angriffe.
- Der Caesar Chiffre ist ein Ersetzungschiffre.

Schlussfolgerung.

- Also ist der Caesar Chiffre anfällig für brute-force Angriffe.

Stimmt das Argument?

Wir können das Argument auf zwei Ebenen diskutieren: Abstrakt

In dem Beispiel wird eine Schlussfolgerung aus gemachten Annahmen gezogen.

Ist das überhaupt korrekt so?

Kann man aus den Annahmen die gemachte Behauptung wirklich folgern?

Logik?

Beispiel

Annahmen.

- Alle Vögel können fliegen.
- Tweety ist ein Vogel.

Schlussfolgerung.

- Also kann Tweety fliegen.

Beispiel

Annahmen.

- Alle Ersetzungschiffren sind anfällig für brute-force Angriffe.
- Der Caesar Chiffre ist ein Ersetzungschiffre.

Schlussfolgerung.

- Also ist der Caesar Chiffre anfällig für brute-force Angriffe.

Stimmt das Argu

Wir können d

In dem Beispi gezogen.

Ist das überha

Kann man au gern?

Abstrakte Argumentation

Annahmen.

- **Wenn** x V **dann** x F.
- x V

Schlussfolgerung.

- x F

Logik?

Beispiel

Annahmen.

- Alle Vögel können fliegen.
- Tweety ist ein Vogel.

Schlussfolgerung.

- Also kann Tweety fliegen.

Beispiel

Annahmen.

- Alle Ersetzungschiffren sind anfällig für brute-force Angriffe.
- Der Caesar Chiffre ist ein Ersetzungschiffre.

Schlussfolgerung.

- Also ist der Caesar Chiffre anfällig für brute-force Angriffe.

Stimmt das Argu

Wir können d

In dem Beispi … gezogen.

Ist das überha

Kann man au … gern?

Abstrakte Argumentation

Annahmen.

- **Wenn** A **dann** B. $(A \to B)$
- A A

Schlussfolgerung.

- B B

Logik?

Logik. Die Logik stellt Methoden bereit, um solche Argumentationsketten untersuchen zu können.

- **Sprache.** Es werden (formale) Sprachen bereitgestellt, mit denen solche Aussagen präzise formuliert werden können.
  - **Beispiele.** Aussagenlogik, Beschreibungslogiken, ...
- **Methoden.** Es werden Methoden entwickelt, um die Korrektheit der Schlussfolgerungen überprüfen zu können.

Abstrakte Argumentation.

Annahmen.

- **Wenn** A **dann** B.
- A

Schlussfolgerung.

- B

Anwendung der Logik: Wissensrepräsentation

Wissensrepräsentation. Eine praktische Anwendung der Logik sind Wissensrepräsentationssysteme.

Solchen Systemen enthalten fachspezifisches Wissen aus einem Anwendungsbereich und erlauben es, daraus neue Folgerungen abzuleiten, Hypothesen zu testen usw.

Einerseits können solche Systeme Fachpersonen unterstützen indem sie gewisse Aufgaben automatisieren. Andererseits machen sie Fachwissen auch außerhalb der jeweiligen Disziplin zugänglich und nutzbar.

Systeme gibt es z.B. zur Diagnoseunterstützung in der Medizin, in der Biologie und vielen anderen Bereichen.

Beispiel.

- Alle Vögel können fliegen.
- Tweety ist ein Vogel.
- Also kann Tweety fliegen.

Wissensrepräsentationssysteme

Wissensrepräsentationssysteme bestehen aus zwei Teilen.

T-Box (terminology box). Hier werden Konzepte und deren Zusammenhänge beschrieben.

Beispiel. **Wenn** x ein Vogel ist, **dann** kann x fliegen.

A-Box (assertion box). Hier werden grundlegende Fakten (Axiome) festgelegt.

Beispiel. „Tweety ist ein Vogel“ „Tweety ist gelb.“

Beschreibungslogiken. Zur Formalisierung der Konzepte und deren Zusammenhänge werden sogenannte Beschreibungslogiken verwendet.

Aus der Wissensbasis können automatisch neue Fakten abgeleitet werden, Hypothesen getestet oder Inkonsistenzen gefunden werden.

Wie das Beispiel zeigt, steht und fällt der Ansatz damit, dass das in der Wissensbasis gespeicherte Wissen korrekt ist.

Beispiel.

- Alle Vögel können fliegen.
- Tweety ist ein Vogel.
- Also kann Tweety fliegen.

2.3 Syntax der Aussagenlogik

Syntax der Aussagenlogik

Definition (Aussagenvariablen).

Wir fixieren eine abzählbar unendliche Menge AVar von Aussagenvariablen, die V_i für alle $i \geq 0$ enthält.

Definition. (Alphabet)

Das *Alphabet* der Aussagenlogik ist

$$\Sigma_{\mathsf{AL}} := \mathsf{AVar} \cup \{\top, \bot, \neg, \vee, \wedge, \rightarrow, \leftrightarrow, (,)\}$$

Definition.

Die Klasse AL der *aussagenlogischen Formeln* wird durch folgende Grammatik definiert:

$$\varphi ::= \top \mid \bot \mid X \in \mathsf{AVar} \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi) \mid (\varphi \leftrightarrow \varphi)$$

Syntax der Aussagenlogik

Die Aussagenlogik ist also induktiv über folgende Regeln definiert:

Basis (Grundmenge).

- $\top, \bot$ sind aussagenlogische Formeln.
- Jede Variable $X \in \mathrm{AVar}$ ist eine aussagenlogische Formel.

$\top, \bot$ und die Variablen werden *atomare Formeln* oder *Atome* genannt.

Induktionsschritt (Regeln).

- Wenn $\varphi \in \mathrm{AL}$ eine Formel ist, dann auch $\neg\varphi \in \mathrm{AL}$
- Wenn $\varphi, \psi \in \mathrm{AL}$ Formeln sind, dann auch

$$(\varphi \vee \psi), \quad (\varphi \wedge \psi), \quad (\varphi \rightarrow \psi), \quad (\varphi \leftrightarrow \psi)$$

$\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ werden *aussagenlogische Verknüpfungen* genannt.

Grammatik. $\varphi ::= \top \mid \bot \mid X \in \mathrm{AVar} \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi) \mid (\varphi \leftrightarrow \varphi)$

Beispiele

Syntaktisch korrekte Formeln.

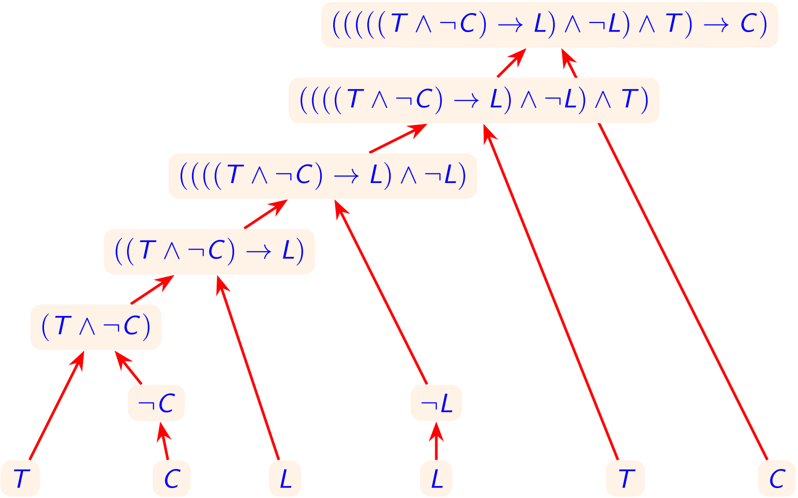
- V_0
- $(T \wedge \neg C)$
- $(((A \wedge B) \wedge C) \vee D)$

Syntaktisch inkorrekte Formeln.

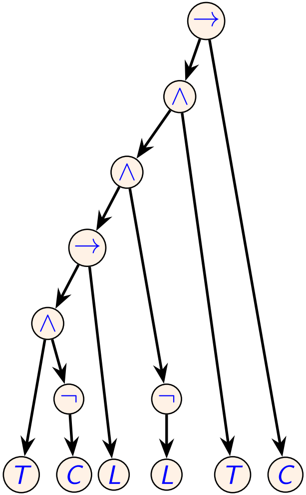
- $(T \& \neg C)$ (falsches Operatorsymbol)
- $A \wedge B$ (fehlende Klammern)
- $(A \wedge B \vee C)$ (fehlende Klammern)
- $\neg(A)$ (zuviele Klammern Klammern)

Grammatik. $\varphi ::= \top \mid \bot \mid X \in \mathrm{AVar} \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi) \mid (\varphi \leftrightarrow \varphi)$

Beispiel: Induktiver Aufbau einer Formel



vgl. Syntaxbaum



Grammatik. $\varphi ::= \top \mid \bot \mid X \in \mathrm{AVar} \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid (\varphi \vee \varphi) \mid (\varphi \rightarrow \varphi) \mid (\varphi \leftrightarrow \varphi)$

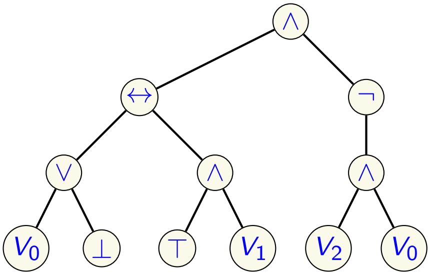
Syntax- oder Ableitungsbäume

Die Struktur einer Formel kann durch ihren *Syntax-* oder *Ableitungsbaum* dargestellt werden.

Der Syntaxbaum der Formel

$$\varphi := (((V_0 \vee \bot) \leftrightarrow (\top \wedge V_1)) \wedge \neg(V_2 \wedge V_0))$$

ist definiert wie folgt:



Präferenzregeln

Präferenzregeln. Um unnötige Klammern zu vermeiden,

- lassen wir die äußersten Klammern weg
- vereinbaren, dass $\neg$ stärker bindet als die anderen Verknüpfungen
- $\wedge, \vee$ binden stärker als $\rightarrow, \leftrightarrow$

Beispiel.

Wir schreiben also $\neg X \wedge Y \rightarrow T$ für $((\neg X \wedge Y) \rightarrow T)$.

Oder $\neg(X \wedge Y \rightarrow T)$ für $\neg((X \wedge Y) \rightarrow T)$.

Aber wir können nicht $X \wedge Y \vee Z$ schreiben.

Präferenzregeln

Notation. Wenn $\Phi = \{\varphi_1, \dots, \varphi_n\} \subseteq \mathsf{AL}$ eine *endliche* Menge von Formeln ist, schreiben wir

- $\bigvee \Phi$ als Abkürzung für $(\varphi_1 \vee \dots \vee \varphi_n)$ die Disjunktion über alle Formeln aus Φ und
- $\bigwedge \Phi$ als Abkürzung für $(\varphi_1 \wedge \dots \wedge \varphi_n)$ die Konjunktion über alle Formeln aus Φ.

Alternative Schreibweise. $\bigwedge_{i=1}^{n} \varphi_i$, $\bigvee_{1 \leq i \leq n} \varphi_i$, usw.

Beispiel.

Wir schreiben also $\bigwedge\{X_i : 1 \leq i \leq n\}$.

Aber wir können nicht $\bigwedge\{X_i : i \geq 1\}$ schreiben, da die Menge nicht endlich ist.

Unterformeln (intuitiv)

Definition. Die Menge $\mathrm{sub}(\varphi)$ der *Unterformeln* einer Formel φ ist die Menge aller *Unterwörter* von φ, die selbst wieder korrekte Formeln sind.

$$\varphi := \big((X \vee Y) \wedge \neg(X \wedge (Y \rightarrow Z))\big)$$

Unterformeln

Definition.

Die Menge $\mathsf{sub}(\varphi)$ der *Unterformeln* einer Formel φ ist induktiv wie folgt definiert:

- Ist φ atomar, dann ist $\mathsf{sub}(\varphi) := \{\varphi\}$.
- Ist $\varphi := \neg\psi$, dann ist $\mathsf{sub}(\varphi) := \{\varphi\} \cup \mathsf{sub}(\psi)$.
- Für alle $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$: Ist $\varphi := (\varphi_1 * \varphi_2)$, dann ist

$$\mathsf{sub}(\varphi) := \{\varphi\} \cup \mathsf{sub}(\varphi_1) \cup \mathsf{sub}(\varphi_2).$$

Wir fassen sub als Funktion $\mathsf{sub} : \mathsf{AL} \rightarrow \mathcal{P}(\mathsf{AL})$ auf, die jeder Formel φ die Menge $\mathsf{sub}(\varphi)$ ihrer Unterformeln zuweist.

Beispiel

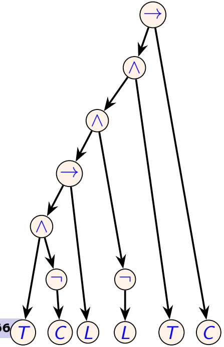
Beispiel. Sei $\varphi := (((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T) \rightarrow C)$.

$$\begin{aligned}\text{sub}(\varphi) := \{ \quad & (((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T) \rightarrow C),\\ & ((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T),\\ & C,\\ & (((T \wedge \neg C) \rightarrow L) \wedge \neg L),\\ & ((T \wedge \neg C) \rightarrow L),\\ & \neg L,\\ & L,\\ & (T \wedge \neg C),\\ & T,\\ & \neg C \ \}.\end{aligned}$$

Definition. Menge $\text{sub}(\varphi)$:

- φ atomar $\rightsquigarrow \text{sub}(\varphi) := \{\varphi\}$.
- $\varphi := \neg\psi \rightsquigarrow \text{sub}(\varphi) := \{\varphi\} \cup \text{sub}(\psi)$.
- Für alle $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$:
  $\varphi := (\varphi_1 * \varphi_2) \rightsquigarrow$
  $\text{sub}(\varphi) := \{\varphi\} \cup \text{sub}(\varphi_1) \cup \text{sub}(\varphi_2)$.

Syntaxbaum



Zusammenfassung

Inhalt.

- Syntax der Aussagenlogik
- Präferenzregeln um die Notation zu vereinfachen
- Definition der Unterformeln einer Formel

2.4 Semantik der Aussagenlogik

Wahrheitsbelegungen

Definition. Die Menge $\mathsf{var}(\varphi)$ der *Variablen einer Formel φ* ist die Menge

$$\mathsf{var}(\varphi) := \mathsf{AVar} \cap \mathsf{sub}(\varphi).$$

Definition.

1. Eine *Wahrheitsbelegung*, oder kurz *Belegung*, ist eine partielle Funktion
$$\beta : \mathsf{AVar} \to \{0, 1\}.$$
2. Eine Belegung β ist eine *Belegung für* eine Formel φ, oder ist *passend für φ*, wenn $\mathsf{var}(\varphi) \subseteq \mathsf{def}(\beta)$.

Intuitiv: 1 steht für *wahr* und 0 für *falsch*.

Semantik der Aussagenlogik

Definition. Per Induktion über die Struktur der Formeln in AL definieren wir eine Funktion $[\![\cdot]\!]^{\cdot}$, die jeder Formel $\varphi \in \mathrm{AL}$ und jeder zu φ passenden Belegung β einen *Wahrheitswert* $[\![\varphi]\!]^\beta \in \{0, 1\}$ zuordnet.

> *Belegung:*
> $\beta : \mathrm{AVar} \to \{0, 1\}$
>
> *passend für φ:*
> $\mathrm{var}(\varphi) \subseteq \mathrm{def}(\beta)$.

Induktionsbasis.

- $[\![\bot]\!]^\beta := 0 \qquad [\![\top]\!]^\beta := 1$
- Für alle $X \in \mathrm{AVar}$ gilt $[\![X]\!]^\beta := \beta(X)$

Induktionsschritt. Für zusammengesetzte Formeln φ ist $[\![\varphi]\!]^\beta$ wie folgt definiert:

- $[\![\neg\varphi]\!]^\beta := 1 - [\![\varphi]\!]^\beta$
- $[\![(\varphi \wedge \psi)]\!]^\beta := \min\{[\![\varphi]\!]^\beta, [\![\psi]\!]^\beta\} \qquad [\![(\varphi \vee \psi)]\!]^\beta := \max\{[\![\varphi]\!]^\beta, [\![\psi]\!]^\beta\}$
- $[\![(\varphi \rightarrow \psi)]\!]^\beta := \begin{cases} 1 & \text{wenn } [\![\varphi]\!]^\beta = 0 \text{ oder } [\![\psi]\!]^\beta = 1 \\ 0 & \text{sonst} \end{cases}$
- $[\![(\varphi \leftrightarrow \psi)]\!]^\beta = 1$ genau dann, wenn $[\![\varphi]\!]^\beta = [\![\psi]\!]^\beta$

Semantik der Aussagenlogik

Definition. Per Induktion über die Struktur der Formeln in AL definieren wir eine Funktion $[\![\cdot]\!]^{\cdot}$, die jeder Formel $\varphi \in \mathrm{AL}$ und jeder zu φ passenden Belegung β einen *Wahrheitswert* $[\![\varphi]\!]^\beta \in \{0,1\}$ zuordnet.

Belegung:
$\beta : \mathrm{AVar} \to \{0,1\}$

passend für φ:
$\mathrm{var}(\varphi) \subseteq \mathrm{def}(\beta)$.

Induktionsb...

- $[\![$...
- Fü...

„tertium non datur"

Der Semantik liegt das Grundprinzip *tertium non datur* zugrunde:
Eine Aussage ist entweder wahr oder falsch.

Induktionsschritt. Für zusammengesetzte Formeln φ ist $[\![\varphi]\!]^\beta$ wie folgt definiert.

- $[\![\neg\varphi]\!]^\beta := 1 - [\![\varphi]\!]^\beta$
- $[\![(\varphi \wedge \psi)]\!]^\beta := \min\{[\![\varphi]\!]^\beta, [\![\psi]\!]^\beta\}$ $\quad [\![(\varphi \vee \psi)]\!]^\beta := \max\{[\![\varphi]\!]^\beta, [\![\psi]\!]^\beta\}$
- $[\![(\varphi \to \psi)]\!]^\beta := \begin{cases} 1 & \text{wenn } [\![\varphi]\!]^\beta = 0 \text{ oder } [\![\psi]\!]^\beta = 1 \\ 0 & \text{sonst} \end{cases}$
- $[\![(\varphi \leftrightarrow \psi)]\!]^\beta = 1$ genau dann, wenn $[\![\varphi]\!]^\beta = [\![\psi]\!]^\beta$

Beispiel

Beispiel. Sei $\varphi := ((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T)$.

- Für $\beta : T \mapsto 1 \quad C \mapsto 1 \quad L \mapsto 0$ gilt $[\![\varphi]\!]^\beta = 1$.

$$\varphi := ((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T)$$

- Für $\beta : T \mapsto 1 \quad C \mapsto 0 \quad L \mapsto 0$ gilt $[\![\varphi]\!]^\beta = 0$.

$$\varphi := ((((T \wedge \neg C) \rightarrow L) \wedge \neg L) \wedge T)$$

Semantik.

- $[\![\bot]\!]^\beta := 0$
- $[\![\top]\!]^\beta := 1$
- Für alle $X \in \mathrm{AVar}$ gilt
  $[\![X]\!]^\beta := \beta(X)$
- $[\![\neg\varphi]\!]^\beta := 1 - [\![\varphi]\!]^\beta$
- $[\![(\varphi \wedge \psi)]\!]^\beta := \min\{[\![\varphi]\!]^\beta, [\![\psi]\!]^\beta\}$
- $[\![(\varphi \vee \psi)]\!]^\beta := \max\{[\![\varphi]\!]^\beta, [\![\psi]\!]^\beta\}$
- $[\![(\varphi \rightarrow \psi)]\!]^\beta :=$
  1 wenn $[\![\varphi]\!]^\beta = 0$ oder $[\![\psi]\!]^\beta = 1$
  0 sonst
- $[\![(\varphi \leftrightarrow \psi)]\!]^\beta = 1$ gdw $[\![\varphi]\!]^\beta = [\![\psi]\!]^\beta$

Eine Bemerkung zur Implikation

Bemerkung. Logische Implikation $\rightarrow$ stimmt nicht immer mit der umgangssprachlichen Verwendung der Implikation überein.

Zum Beispiel wird keine *Kausalität* impliziert.

$\varphi \rightarrow \psi$ heißt einfach, dass wann immer φ wahr ist, auch ψ wahr sein muss.

Insbesondere, wenn φ falsch ist, dann ist $\varphi \rightarrow \psi$ als Aussage wahr.

Über die Wahl der Verknüpfungen

Unsere Wahl der Verknüpfungen $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ für die Aussagenlogik spiegelt unsere Verwendung in natürlicher Sprache wieder, ist aber zu bestimmtem Grad willkürlich.

Durch die Definition einer Wahrheitstafel können wir auch andere Verknüpfungen und somit auch andere "Aussagenlogiken" definieren.

Beispiel. *Exklusives Oder* $\varphi \oplus \psi$

Exklusives Oder

$[\![\varphi]\!]^\beta$	$[\![\psi]\!]^\beta$	$[\![\varphi \oplus \psi]\!]^\beta$
0	0	0
0	1	1
1	0	1
1	1	0

Intuitiv: $\varphi \oplus \psi$ bedeutet *"entweder φ oder ψ"*

Notation

Notation. Um die Lesbarkeit zu erhöhen, schreiben wir:

- Belegungen: $\beta, \gamma, \ldots$
- Formeln: $\varphi, \psi, \varphi' \ldots$
- Mengen von Formeln: $\Phi, \Psi, \ldots$
- Wir werden auch $X, Y, \ldots$ für Variablen verwenden

Das Koinzidenz Lemma

Lemma (Koinzidenzlemma).

Sei $\varphi \in \mathrm{AL}$ eine Formel und seien β, β' Belegungen so dass

$$\beta(X) = \beta'(X) \qquad \text{für alle } X \in \mathrm{var}(\varphi).$$

Dann gilt $[\![\varphi]\!]^{\beta} = [\![\varphi]\!]^{\beta'}$.

Zusammenfassung

Die Semantik einer Logik bestimmt die Bedeutung der Formeln.

Variablen in der Aussagenlogik sind Platzhalter für Aussagen, die wahr oder falsch sein können.

Eine (passende) *Belegung* weist den Variablen einer Formel Wahrheitswerte zu und bestimmt damit die *Umgebung* in der die Formel ausgewertet wird.

Eine Formel φ hat unter jeder passenden Belegung β einen eindeutigen Wahrheitswert $[\![\varphi]\!]^{\beta}$.

2.5 Abstrakte Beispiele

Ein abstraktes Beispiel

Beispiel. Sei $\varphi := M \wedge (B \wedge (\neg F \vee G))$.

Es gilt $\text{var}(\varphi) := \{M, B, F, G\}$.

Eine Belegung β *passt* also zu φ, wenn $\{M, B, F, G\} \subseteq \text{def}(\beta)$.

Seien β_1, β_2 definiert durch

	M	B	F	G
β_1	1	1	1	0
β_2	1	1	0	0

Dann gilt:

$[\![M \wedge (B \wedge (\neg F \vee G))]\!]^{\beta_1} = 0$ aber $[\![M \wedge (B \wedge (\neg F \vee G))]\!]^{\beta_2} = 1$.

Semantik.
- $[\![\bot]\!]^\beta := 0$
- $[\![\top]\!]^\beta := 1$
- Für alle $X \in \text{AVar}$ gilt
  $[\![X]\!]^\beta := \beta(X)$
- $[\![\neg\varphi]\!]^\beta := 1 - [\![\varphi]\!]^\beta$
- $[\![(\varphi \wedge \psi)]\!]^\beta := \min\{[\![\varphi]\!]^\beta, [\![\psi]\!]^\beta\}$
- $[\![(\varphi \vee \psi)]\!]^\beta := \max\{[\![\varphi]\!]^\beta, [\![\psi]\!]^\beta\}$
- $[\![(\varphi \rightarrow \psi)]\!]^\beta :=$
  1 wenn $[\![\varphi]\!]^\beta = 0$ oder $[\![\psi]\!]^\beta = 1$
  0 sonst
- $[\![(\varphi \leftrightarrow \psi)]\!]^\beta = 1$ gdw $[\![\varphi]\!]^\beta = [\![\psi]\!]^\beta$

Belegung:
$\beta : \text{AVar} \rightarrow \{0, 1\}$

passend für φ:
$\text{var}(\varphi) \subseteq \text{def}(\beta)$.

Ein abstraktes Beispiel

Beispiel 2. Sei $\varphi_2 := (X \rightarrow Y) \leftrightarrow (\neg X \vee Y)$. Es gilt $\mathrm{var}(\varphi_2) := \{X, Y\}$.

Um die Belegungen zu finden, die φ erfüllen, kann eine Wahrheitstafel benutzt werden.

X	Y	$(X \rightarrow Y)$	$(\neg X \vee Y)$	φ_2
0	0	1	1	1
0	1	1	1	1
1	0	0	0	1
1	1	1	1	1

Jede zu φ_2 passende Belegung erfüllt die Formel.

Ein abstraktes Beispiel

Beispiel 3. Sei

$$\varphi_3 := (X \leftrightarrow \neg Y) \wedge ((X \wedge P) \vee (\neg X \wedge W)) \wedge (P \rightarrow Y) \wedge (W \rightarrow \neg Y)$$

Es gilt $\mathrm{var}(\varphi_3) := \{X, Y, P, W\}$.

Um die Belegungen zu finden, die φ_3 erfüllen, kann eine Wahrheitstafel benutzt werden.

X	Y	P	W	$X \leftrightarrow \neg Y$	$(X \wedge P) \vee (\neg X \wedge W)$	$P \rightarrow Y$	$W \rightarrow \neg Y$	φ_3
1	1	1	1	0	1	1	1	0
1	0	1	0	1	1	0	1	0
1	0	0	0	1	0	1	1	0
1	0	0	1	1	0	1	1	0

Die Formel φ_3 hat keine erfüllende Belegung.

Ein abstraktes Beispiel

Drei Beispiele.

1. $\varphi_1 := M \wedge (B \wedge (\neg F \vee G))$.
   Die Formel wurde durch β_2 erfüllt, durch β_1 aber nicht.
2. $\varphi_2 := (X \rightarrow Y) \leftrightarrow (\neg X \vee Y)$.
   Die Formel wurde durch jede passende Belegung erfüllt.
3. $\varphi_3 := (X \leftrightarrow \neg Y) \wedge ((X \wedge P) \vee (\neg X \wedge W)) \wedge (P \rightarrow Y) \wedge (W \rightarrow \neg Y)$.
   Die Formel wurde durch keine Belegung erfüllt.

Erfüllbarkeit und Allgemeingültigkeit

Definition. Sei $\varphi \in \mathsf{AL}$ eine Formel.

1. Eine zu φ passende Belegung β *erfüllt* φ, oder ist ein *Modell* von φ, wenn $[\![\varphi]\!]^\beta = 1$.

   Wir schreiben $\beta \models \varphi$.

2. φ ist *erfüllbar*, wenn es eine Belegung β gibt, die φ erfüllt. Anderenfalls ist φ *unerfüllbar*.

3. φ ist *allgemeingültig*, oder eine *Tautologie*, wenn jede zu φ passende Belegung φ erfüllt.

Erfüllbarkeit und Allgemeingültigkeit

Definition. Sei $\varphi \in \mathrm{AL}$ eine Formel.

1. Eine zu φ passende Belegung β *erfüllt* φ, oder ist ein *Modell* von φ, wenn $[\![\varphi]\!]^\beta = 1$.
   Wir schreiben $\beta \models \varphi$.
2. φ ist *erfüllbar*, wenn es eine Belegung β gibt, die φ erfüllt.
   Anderenfalls ist φ *unerfüllbar*.
3. φ ist *allgemeingültig*, oder eine *Tautologie*, wenn jede zu φ passende Belegung φ erfüllt.

$\varphi_1 := M \wedge (B \wedge (\neg F \vee G))$

$\varphi_3 := (X \leftrightarrow \neg Y) \wedge ((X \wedge P) \vee (\neg X \wedge W)) \wedge (P \rightarrow Y) \wedge (W \rightarrow \neg Y)$

$\varphi_2 := (X \rightarrow Y) \leftrightarrow (\neg X \vee Y)$

2.6 Beispiele: Grundlagen des formalen Beweisens

Beispiel

Beispiel. Betrachten wir folgende Argumentation.

Bekannt:

Wenn ein Graph zusammenhängend ist und keinen Kreis enthält, dann enthält er $n-1$ Kanten.

Angenommen, ein gegebener Graph G

- enthält $> n-1$ Kanten und
- ist zusammenhängend.

Dann enthält G einen Kreis.

Ist das Argument *gültig*?

Formalisierung des Beispiels

Beispiel. Ist das Argument *gültig*?

Bekannt:

Wenn ein $\underbrace{\text{Graph zusammenhängend}}_{A}$ ist und keinen $\underbrace{\text{Kreis enthält}}_{B}$, dann enthält er $\underbrace{n-1 \text{ Kanten}}_{C}$.

Angenommen, ein gegebener Graph G

- enthält $> n-1$ Kanten und
- ist zusammenhängend.

Dann enthält G einen Kreis.

Formalisierung.

1. $A \wedge \neg B \to C$
2. $\neg C$
3. A

Daraus soll B folgen.

Wir müssen also entscheiden, ob die Formel $((A \wedge \neg B \to C) \wedge \neg C \wedge A) \to B$ allgemeingültig ist.

Beispiel

Beispiel. Sie kann nicht zu hause sein, da sie an Bord oder zuhause ist und ich gerade gehört habe, dass sie an Bord ist.

Ist das Argument *gültig*?

Was können wir formal beweisen?

- Klar formulierte Aussagen, die entweder richtig oder falsch sind.

  Die Bedeutung aller verwendeten Ausdrücke muss bekannt sein.

  Ebenso das vorausgesetzte Hintergrundwissen.

- Natürliche Sprache ist dafür nicht gut geeignet.

- Wir werden daher formale Sprachen, oder Logiken, verwenden, in denen alle Ausdrücke formal und vollständig definiert sind.

- Ziel ist es, allgemeine Regeln für korrektes Schließen herleiten zu können, möglichst sogar automatisch.

2.7 Beispiele: Logik und Algorithmen I

Sudokus leicht gemacht

Aussagenlogik als algorithmisches Hilfsmittel

Sudoku

Sudoku. Betrachten wir das Spiel Sudoku.

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

Regeln.

Fülle Felder mit Zahlen aus $1, \ldots, 9$, so dass jede Zahl genau einmal in

- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Aufgabe ist es, die fehlenden Positionen so mit den Zahlen $1, ..., 9$ zu füllen, dass in jeder Zeile und jeder Spalte und in jedem Block jede der Zahlen $1, ..., 9$ genau einmal vorkommt.

Sudoku

Sudoku. Betrachten wir das Spiel Sudoku.

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

Regeln.

Fülle Felder mit Zahlen aus $1, \dots, 9$, so dass jede Zahl genau einmal in

- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Aufgabe ist es, die fehlenden Positionen so mit den Zahlen $1, \dots, 9$ zu füllen, dass in jeder Zeile und jeder Spalte und in jedem Block jede der Zahlen $1, \dots, 9$ genau einmal vorkommt.

Sudoku

Ziel. Sudokus mit Hilfe der Aussagenlogik lösen.

Wir wollen also zu einem gegebenen Sudoku S eine Formel φ_S konstruieren, so dass die erfüllenden Belegungen von φ_S genau den Lösungen des Sudokus entsprechen.

Es muss also einen inhaltlichen Zusammenhang zwischen einer erfüllenden Belegung und einer Beschriftung des Sudokus geben.

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

Herangehensweise.
Wir müssen uns als erstes überlegen, welche Variablen wir verwenden wollen und was die Variablen bedeuten sollen.

Regeln.
Fülle Felder mit $1, \dots, 9$ s.d. jede Zahl genau einmal in
- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Idee 1. Für jede Position (i, j) eine Variable $X_{i,j}$.
Die Variablen werden mit der Zahl belegt, die an der Position stehen soll.

Problem. Variablen können nur wahr oder falsch werden.

Sudoku

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

Ziel. Konstruiere zu gegebenen Sudoku S eine Formel φ_S, so dass: erfüllenden Belegungen von φ_S entsprechen Lösungen des Sudokus

Herangehensweise.

Wir müssen uns als erstes überlegen, welche Variablen wir verwenden wollen und was die Variablen bedeuten sollen.

Variablen. Können nur wahr oder falsch werden.

Was soll es bedeuten, wenn eine Belegung β eine Variable X mit 1 belegt?

Idee 2. Für jede Position (i, j) und jede mögliche Beschriftung $c \in \{1, \dots, 9\}$ führen wir eine Variable $X^c_{i,j}$ ein.

$X^c_{i,j}$ wird wahr, wenn an der Stelle (i, j) die Zahl c steht.

Regeln.

Fülle Felder mit $1, \dots, 9$ s.d. jede Zahl genau einmal in

- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Sudoku

Ziel. Konstruiere zu gegebenen Sudoku S eine Formel φ_S, so dass:
erfüllenden Belegungen von φ_S entsprechen Lösungen des Sudokus

Variablen. Für jede Position (i, j) und jede mögliche Beschriftung $c \in \{1, \dots, 9\}$ führen wir eine Variable $X_{i,j}^c$ ein.

$X_{i,j}^c$ wird wahr, wenn an der Stelle (i, j) die Zahl c steht.

Belegungen und Lösungen.

Aus Lösung L des Sudokus können wir Belegung β_L konstruieren:

$\beta_L(X_{i,j}^c) = 1$ gdw. L das Feld (i, j) mit 1 beschriftet.

Umgekehrt, wollen wir aus einer Belegung β eine Beschriftung L_β konstruieren:

L_β beschriftet die Position (i, j) mit der Zahl c gdw. $\beta(X_{i,j}^c) = 1$.

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

Regeln.
Fülle Felder mit $1, \dots, 9$ s.d. jede Zahl genau einmal in
- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Sudoku

Variablen. Für jede Position (i,j) und jede mögliche Beschriftung $c \in \{1,\dots,9\}$ führen wir eine Variable $X^c_{i,j}$ ein.

$X^c_{i,j}$ wird **wahr**, wenn an der Stelle (i,j) die Zahl c steht.

Von Belegungen β zu Lösungen L_β.

L_β beschriftet die Position (i,j) mit der Zahl c gdw. $\beta(X^c_{i,j}) = 1$.

Dazu muss φ_S sicherstellen, dass es für alle (i,j) genau ein $c \in \{1,\dots,9\}$ gibt, so dass $\beta_S(X^c_{i,j}) = 1$.

Sei

$$\varphi_{beschr} := \bigwedge_{1\leq i,j\leq 9} \bigvee_{c=1}^{9} \Big(X^c_{i,j} \wedge \bigwedge_{\substack{d\neq c,\\ d\in\{1,\dots,9\}}} \neg X^d_{i,j} \Big)$$

Erfüllende Belegungen β von φ_{beschr} entsprechen genau Beschriftungen der Sudoku-Felder mit Zahlen aus $\{1,\dots,9\}$.

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

Regeln.

Fülle Felder mit $1,\dots,9$ s.d. jede Zahl genau einmal in
- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Sudoku

Variablen. Für jede Position (i, j) und jede mögliche Beschriftung $c \in \{1, \dots, 9\}$ führen wir eine Variable $X^c_{i,j}$ ein.

$X^c_{i,j}$ wird wahr, wenn an der Stelle (i, j) die Zahl c steht.

Belegungen und Beschriftungen. Die Formel φ_{beschr} garantiert uns, dass erfüllende Belegungen β genau möglichen Beschriftungen L_β der Sudoku-Felder entsprechen.

Aber: L_β muss keine gültige Lösung des gegebenen Sudokus sein!

Wir müssen noch folgendes sicherstellen:

- Die Beschriftung L_β entspricht den vorgegebenen Feldern.
- Die Beschriftung L_β erfüllt die Sudoku-Regeln.

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

Regeln.
Fülle Felder mit $1, \dots, 9$ s.d. jede Zahl genau einmal in
- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Formalisieren der vorausgefüllten Felder

Variablen. Für jede Position (i, j) und jede mögliche Beschriftung $c \in \{1, \dots, 9\}$ führen wir eine Variable $X^c_{i,j}$ ein.

$\beta(X^c_{i,j}) = 1$ gdw. L_β an die Stelle (i, j) die Zahl c schreibt.

Vorgegebene Anfangsbeschriftung.

"Die Beschriftung L_β entspricht den vorgegebenen Feldern."

$$\varphi^S_{anfang} := \bigwedge\{X^c_{i,j} : \text{ die Pos. } (i, j) \text{ ist mit } c \text{ vorausgefüllt}\}$$

Im Beispiel oben: $\varphi^S_{anfang} := X^6_{5,1} \wedge X^8_{6,1} \wedge X^1_{7,1} \wedge \dots$

Eine Belegung β die $\varphi_{beschr} \wedge \varphi^S_{anfang}$ erfüllt entspricht also einer Beschriftung L_β, die dem gegebenen Sudoku entspricht.

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

Regeln.

Fülle Felder mit $1, \dots, 9$ s.d. jede Zahl genau einmal in

- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Formalisieren der Sudoku-Regeln

Variablen. Für jede Position (i,j) und jede mögliche Beschriftung $c \in \{1,\ldots,9\}$ führen wir eine Variable $X^c_{i,j}$ ein.

$\beta(X^c_{i,j}) = 1$ gdw. L_β an die Stelle (i,j) die Zahl c schreibt.

Die Sudoku-Regeln. "Die Beschriftung L_β erfüllt die Sudoku-Regeln"

- In jeder Zeile kommt jede Zahl aus $\{1,\ldots,9\}$ genau einmal vor.
  Für alle Zeilen i:
  für verschiedene Spalten $j \neq j'$ und Zahlen c gilt nicht:
  Position (j,i) und Position (j',i) ist mit c beschriftet.

$$\varphi_{zeile} := \bigwedge_{i=1}^{9} \bigwedge_{1\leq j<j'\leq 9} \bigwedge_{c=1}^{9} \neg(X^c_{j,i} \wedge X^c_{j',i}).$$

- In jeder Spalte kommt jede Zahl aus $\{1,\ldots,9\}$ genau einmal vor.
- In jedem Block kommt jede Zahl aus $\{1,\ldots,9\}$ genau einmal vor.

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

Regeln.
Fülle Felder mit $1,\ldots,9$ s.d. jede Zahl genau einmal in
- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Formalisieren der Sudoku-Regeln

Regel für Zeilen. "In jeder Zeile kommt jedes $c \in \{1, \dots, 9\}$ genau einmal vor."

Für alle Zeilen i und Spalten $j \neq j'$ und Zahlen c gilt **nicht**:

Position (j, i) und Position (j', i) ist mit c beschriftet.

$$\varphi_{zeile} := \bigwedge_{i=1}^{9} \ \bigwedge_{1 \leq j < j' \leq 9} \ \bigwedge_{c=1}^{9} \ \neg\left(X_{j,i}^{c} \wedge X_{j',i}^{c}\right).$$

Regel für Spalten. "In jeder Spalte kommt jede Zahl aus $\{1, \dots, 9\}$ genau einmal vor."

$$\varphi_{spalte} := \bigwedge_{i=1}^{9} \ \bigwedge_{1 \leq j < j' \leq 9} \ \bigwedge_{c=1}^{9} \ \neg\left(X_{i,j}^{c} \wedge X_{i,j'}^{c}\right).$$

Regel für Blöcke. "In jedem Block kommt jede Zahl aus $\{1, \dots, 9\}$ genau einmal vor."

$$\varphi_{block} :=$$
$$\bigwedge \left\{ \neg\left(X_{i,j}^{c} \wedge X_{i',j'}^{c}\right) : (i,j) \neq (i',j'),\ \left\lfloor \frac{i-1}{3} \right\rfloor = \left\lfloor \frac{i'-1}{3} \right\rfloor \text{ und } \left\lfloor \frac{j-1}{3} \right\rfloor = \left\lfloor \frac{j'-1}{3} \right\rfloor \right\}$$

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

Regeln.
Fülle Felder mit $1, \dots, 9$ s.d. jede Zahl genau einmal in
- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Formalisierung von Sudoku in der Aussagenlogik

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

Variablen. Für jede Position (i, j) und jede mögliche Beschriftung $c \in \{1, \ldots, 9\}$ führen wir eine Variable $X_{i,j}^c$ ein.

Die Formel φ_S. Sei S ein gegebenes Sudoku.

$$\varphi_S := \varphi_{anfang}^S \wedge \varphi_{beschr} \wedge \varphi_{zeile} \wedge \varphi_{spalte} \wedge \varphi_{block}$$

Dann gilt: Jede Belegung β, die φ_S erfüllt, induziert eine gültige Lösung L_β des Sudokus S.

L_β beschriftet die Position (i, j) mit c gdw. $\beta(X_{i,j}^c) = 1$.

Jede gültige Lösung L des Sudokus S induziert eine Belegung β_L, die φ_S erfüllt.

$\beta(X_{i,j}^c) = 1$ gdw. L_β die Position (i, j) mit c beschriftet.

Lemma. φ_S ist genau dann erfüllbar, wenn das Sudoku S lösbar ist.

Regeln.
Fülle Felder mit $1, \ldots, 9$ s.d. jede Zahl genau einmal in
- jeder Zeile
- jeder Spalte
- jedem Block

vorkommt.

Zusammenfassung

Wie wir am Beispiel des Sudokus gesehen haben, können Probleme dieser Art, sogenannte *constraint satisfaction problems*, recht leicht in der Aussagenlogik formalisiert werden.

Die Belegungen der Aussagenvariablen entsprechen dabei potentiellen Lösungen des Sudokus.

Erfüllende Belegungen entsprechen dann genau den korrekten Lösungen.

Wir brauchen also nur noch schnelle Verfahren, um Formeln der Aussagenlogik auf Erfüllbarkeit testen zu können.

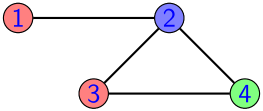
$\leadsto$ SAT-Löser und der DPLL Algorithmus

2.8 Beispiele: Effizientes Lösen NP-vollständiger Probleme

3-Färbbarkeit

Definition (3-Färbbarkeit). Ein Graph G ist 3-*färbbar*, wenn es eine Funktion $c : V(G) \to \{C_1, C_2, C_3\}$ gibt, so dass $c(u) \neq c(v)$ für alle Kanten $\{u, v\} \in E(G)$.

Wir nennen $c : V(G) \to \{1, 2, 3\}$ eine 3-*Färbung* von G.



Das Problem 3-COL. Das zugehörige Entscheidungsproblem

3-COL	
Eingabe.	Graph G.
Problem.	Entscheide, ob G 3-färbbar ist.

ist NP-vollständig.

Lösung durch die Aussagenlogik. Wir werden im Verlauf dieses Moduls sehen, wie das Problem mit Hilfe der Aussagenlogik gelöst werden kann.

Schritt 1: Formalisierung. Konstruiere aus einem Graph G eine Formel $\varphi_G \in \mathrm{AL}$ die genau dann erfüllbar ist, wenn G 3-färbbar ist.

Eine Anwendung: 3-Färbbarkeit

Lemma.

Zu jedem endlichen Graph G können wir in polynomieller Zeit eine Formel φ_G konstruieren, so dass G genau dann 3-färbbar ist, wenn φ_G erfüllbar ist.

Grundidee.

3-Färbbarkeit. Wir suchen eine Funktion $c : V(G) \to \{1, 2, 3\}$, die die Knoten von G korrekt färbt.

Erfüllbarkeit. Wir suchen eine Belegung $\beta : \text{var}(\varphi) \to \{0, 1\}$, die die Formel erfüllt.

Ziel. Konstruiere aus dem Graph G eine Formel φ_G, so dass

- *Belegungen* von $\text{var}(\varphi)$ den möglichen *Beschriftungen* der Knoten mit Farben aus $\{1, 2, 3\}$ entsprechen und
- *erfüllende Belegungen* genau den *korrekten 3-Färbungen* von G entsprechen.

Definition.

$G = (V, E)$ *3-färbbar*, wenn

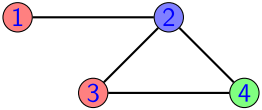
3-Färbung

$c : V \to \{C_1, C_2, C_3\}$

existiert, so dass

$c(u) \neq c(v)$

für alle $\{u, v\} \in E$.



Eine Anwendung: 3-Färbbarkeit

Lemma.

Zu jedem endlichen Graph G können wir in polynomieller Zeit eine Formel φ_G konstruieren, so dass G genau dann 3-färbbar ist, wenn φ_G erfüllbar ist.

Variablen. $X_{u,i}$ für alle $u \in V$ und $1 \leq i \leq 3$.

Intention. Variable $X_{u,i}$ wird wahr, wenn Knoten u mit C_i gefärbt wird.

Färbungen vs. Belegungen.

Eine Funktion $c : V \to \{C_1, C_2, C_3\}$ entspricht der Belegung β_c mit $\beta_c(X_{u,i}) = 1$ gdw. $c(u) = C_i$.

Plan. Belegung β entspricht Funktion $c : V \to \{C_1, C_2, C_3\}$ mit

$$c(u) = C_i \text{ gdw. } \beta(X_{u,i}) = 1.$$

Ist aber falsch!

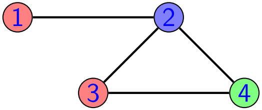
Definition.
$G = (V, E)$ *3-färbbar*, wenn

3-Färbung $c : V \to \{C_1, C_2, C_3\}$

existiert, so dass

$c(u) \neq c(v)$

für alle $\{u, v\} \in E$.



Beweis des Lemmas

Färbungen vs. Belegungen.

$c : V \to \{C_1, C_2, C_3\}$ entspricht β_c mit $\beta_c(X_{u,i}) = 1$ gdw. $c(u) = C_i$.

β entspricht $c : V \to \{C_1, C_2, C_3\}$ mit $c(u) = C_i$ gdw. $\beta(X_{u,i}) = 1$.

Bedingungen an Färbungsfunktion.

$$\varphi_1 := \bigwedge \{ (X_{u,1} \vee X_{u,2} \vee X_{u,3}) \quad : \quad u \in V \}$$

„Jeder Knoten erhält eine Farbe"

$$\varphi_2 := \bigwedge \{ \neg(X_{u,i} \wedge X_{u,j}) \quad : \quad u \in V, 1 \leq i \neq j \leq 3 \}$$

„Kein Knoten erhält zwei Farben"

$$\varphi_3 := \bigwedge \{ \bigwedge_{i=1}^{3} \neg(X_{u,i} \wedge X_{v,i}) \quad : \quad \{u, v\} \in E \}$$

„Keine monochromatische Kante"

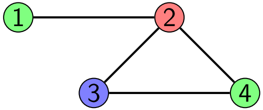
Beobachtung. Belegungen β mit $\beta \models \varphi_1 \wedge \varphi_2 \wedge \varphi_3$ entsprechen genau den gültigen Färbungen.

Definition.
$G = (V, E)$ *3-färbbar*, wenn
3-Färbung $c : V \to \{C_1, C_2, C_3\}$
existiert, so dass
$c(u) \neq c(v)$
für alle $\{u, v\} \in E$.

Variablen.
$X_{u,i}$ für $u \in V$, $1 \leq i \leq 3$.

Bedeutung.
$\beta_c(X_{u,i}) = 1$ gdw $c(u) = C_i$.

Beispiel für 3-Färbbarkeit



Formeln.

$\varphi_1 := \bigwedge \{ (X_{u,1} \vee X_{u,2} \vee X_{u,3}) : u \in V \}$

„Jeder Knoten erhält eine Farbe“

$\varphi_2 := \bigwedge \{ \neg(X_{u,i} \wedge X_{u,j}) : u \in V, 1 \leq i \neq j \leq 3 \}$

„Kein Knoten erhält zwei Farben“

$\varphi_3 := \bigwedge \{ \bigwedge_{i=1}^{3} \neg(X_{u,i} \wedge X_{v,i}) : \{u,v\} \in E \}$

„Keine monochromatische Kante“

Formel $\Phi_G := \varphi_1 \wedge \varphi_2 \wedge \varphi_3$

$$\varphi_1 := \bigwedge \{ (X_{1,1} \vee X_{1,2} \vee X_{1,3}), \quad (X_{2,1} \vee X_{2,2} \vee X_{2,3}), \quad (X_{2,1} \vee X_{2,2} \vee X_{2,3}), \quad (X_{2,1} \vee X_{2,2} \vee X_{2,3}) \}$$

$$\varphi_2 := \bigwedge \{ (\neg(X_{1,1} \wedge X_{1,2}) \wedge \neg(X_{1,1} \wedge X_{1,3}) \wedge \neg(X_{1,2} \wedge X_{1,3})), \dots \}$$

$$\varphi_3 := \bigwedge \{ (\neg(X_{1,1} \wedge X_{2,1}) \wedge \neg(X_{1,2} \wedge X_{2,2}) \wedge \neg(X_{1,3} \wedge X_{2,3})), \dots \}$$

Erfüllende Belegungen vs. Färbungen.

	$X_{1,1}$	$X_{1,2}$	$X_{1,3}$	$X_{2,1}$	$X_{2,2}1$	$X_{2,3}$	$X_{3,1}$	$X_{3,2}1$	$X_{3,3}$	$X_{4,1}$	$X_{4,2}1$	$X_{4,3}$
β_1	1	0	0	0	0	1	1	0	0	0	1	0
β_2	0	1	0	1	0	0	0	0	1	0	1	0

Beweis des Lemmas

Formel. $\varphi_G := \varphi_1 \wedge \varphi_2 \wedge \varphi_3$

$\varphi_1 := \bigwedge \{ (X_{u,1} \vee X_{u,2} \vee X_{u,3}) : u \in V \}$

„Jeder Knoten erhält eine Farbe“

$\varphi_2 := \bigwedge \{ \neg (X_{u,i} \wedge X_{u,j}) : u \in V, 1 \leq i \neq j \leq 3 \}$

„Kein Knoten erhält zwei Farben“

$\varphi_3 := \bigwedge \{ \bigwedge_{i=1}^{3} \neg (X_{u,i} \wedge X_{v,i}) : \{u, v\} \in E \}$

„Keine monochromatische Kante“

Behauptung. φ_G ist genau dann erfüllbar, wenn G 3-färbbar ist.

Beweis der Rückrichtung. Sei $c : V \to \{C_1, C_2, C_3\}$ eine 3-Färbung von G.

Definiere β_c durch: $\beta_c(X_{u,i}) = 1$ gdw. $c(u) = C_i$.

Dann gilt:

- $\beta_c \models \varphi_1$, denn für alle $u \in V$:

  wenn $c(u) = C_i$, dann $\beta_c(X_{u,i}) = 1$ und somit $\beta_c \models (X_{u,1} \vee X_{u,2} \vee X_{u,3})$.
- $\beta_c \models \varphi_2$, denn für alle $u \in V$ gibt es nur ein i mit $c(u) = C_i$.

  Daher gilt $\beta_c(X_{u,i}) = 1$ auch nur für einen Index i.

  Für $i \neq j$ kann daher $(X_{u,i} \wedge X_{u,j})$ niemals in β_c wahr werden.
- $\beta_c \models \varphi_3$, folgt genauso.

Also gilt $\beta_c \models \varphi_G$.

Beweis des Lemmas

Formel. $\varphi_G := \varphi_1 \wedge \varphi_2 \wedge \varphi_3$

$\varphi_1 := \bigwedge \{(X_{u,1} \vee X_{u,2} \vee X_{u,3}) : u \in V\}$

„*Jeder Knoten erhält eine Farbe*“

$\varphi_2 := \bigwedge\{\neg(X_{u,i} \wedge X_{u,j}) : u \in V, 1 \leq i \neq j \leq 3\}$

„*Kein Knoten erhält zwei Farben*“

$\varphi_3 := \bigwedge \{ \bigwedge_{i=1}^{3} \neg(X_{u,i} \wedge X_{v,i}) : \{u,v\} \in E\}$

„*Keine monochromatische Kante*“

Behauptung. φ_G ist genau dann erfüllbar, wenn G 3-färbbar ist.

Beweis der Rückrichtung. Sei $c : V \to \{C_1, C_2, C_3\}$ eine 3-Färbung von G.

Definiere β_c durch: $\beta_c(X_{u,i}) = 1$ gdw. $c(u) = C_i$.

Also gilt $\beta_c \models \varphi_G$.

Hinrichtung. Umgekehrt, sei $\beta \models \varphi_G$ ein erfüllende Belegung.

Da β die Formeln φ_1 und φ_2 erfüllt, gibt es für jeden Knoten $u \in V$ genau ein $i_u \in \{1,2,3\}$ mit $\beta(X_{u,i_u}) = 1$.

Definiere eine Färbung $c : V \to \{C_1, C_2, C_3\}$ durch $c(u) := C_{i_u}$.

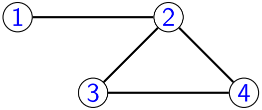
Da β die Formeln in 3) erfüllt, ist c eine gültige 3-Färbung. $\dashv$

Anwendung: 3-Färbbarkeit

Definition. Ein Graph G ist *3-färbbar*, wenn es eine Funktion $c : V \to \{C_1, C_2, C_3\}$ gibt, so dass $c(u) \neq c(v)$ für alle Kanten $\{u, v\} \in E$.

Lemma.

Zu jedem endlichen Graph G können wir in polynomieller Zeit eine Formel φ_G konstruieren, so dass G genau dann 3-färbbar ist, wenn φ_G erfüllbar ist.



Anwendung: 3-Färbbarkeit

Definition. Ein Graph G ist *3-färbbar*, wenn es eine Funktion $c : V \to \{C_1, C_2, C_3\}$ gibt, so dass $c(u) \neq c(v)$ für alle Kanten $\{u, v\} \in E$.

Lemma.

Zu jedem endlichen Graph G können wir in polynomieller Zeit eine Formel φ_G konstruieren, so dass G genau dann 3-färbbar ist, wenn φ_G erfüllbar ist.

Moderne SAT-Löser können Erfüllbarkeit sehr effizient entscheiden (meistens).

Somit erhalten wir einen einfachen und effizienten Algorithmus für `3-COL`.

Wie das geht, werden wir in Woche 6 genauer besprechen.

Formalisieren algorithmischer Probleme in Aussagenlogik

Teilmengen einer Menge. Sei M eine Menge mit n Elementen.

Teilmengen $N \subseteq M$ können wir wie folgt durch Belegungen kodieren:

- Wir führen für jedes Element $m \in M$ eine Variable X_m ein.
- Eine Belegung $\beta : \{X_m : m \in M\} \to \{0, 1\}$ entspricht dann genau der Teilmenge $M_\beta = \{m \in M : \beta(X_m) = 1\}$.

Kodieren von Funktionen. Seien M und N endliche Mengen.

Funktionen $f : M \to N$ können wir wie folgt durch Belegungen β_f kodieren.

Variablen. $V := \{X_{m,n} : m \in M, n \in N\}$

Belegung. $\beta : V \to \{0, 1\}$ entspricht $f_\beta : M \to \mathcal{P}(N)$ mit $f_\beta(m) = \{n \in N : \beta(X_{m,n}) = 1\}$.

Bedingungen. Betrachte Formel $\varphi := \bigwedge\{\bigvee_{n \in N} \left(X_{m,n} \wedge \bigwedge_{n' \neq n} \neg X_{m,n'}\right) : m \in M\}$.

Wenn $\beta \models \varphi$, dann entspricht β einer Funktion $f_\beta : M \to N$.

Zusammenfassung

Reduktion auf SAT.

- Bestimmte algorithmische Probleme lassen sich recht einfach in der Aussagenlogik formalisieren.
- Dies gilt z.B. für viele NP-vollständige Probleme aus der Graphentheorie.
- Wir erhalten damit ein sehr allgemeines und mächtiges Werkzeug um schwere algorithmische Probleme in der Praxis effizient zu lösen.

Formalisierungstricks.

- Welcher Teil des Problems soll den erfüllenden Belegungen entsprechen?
- Kodierungstricks helfen bei der konkreten Formalisierung.