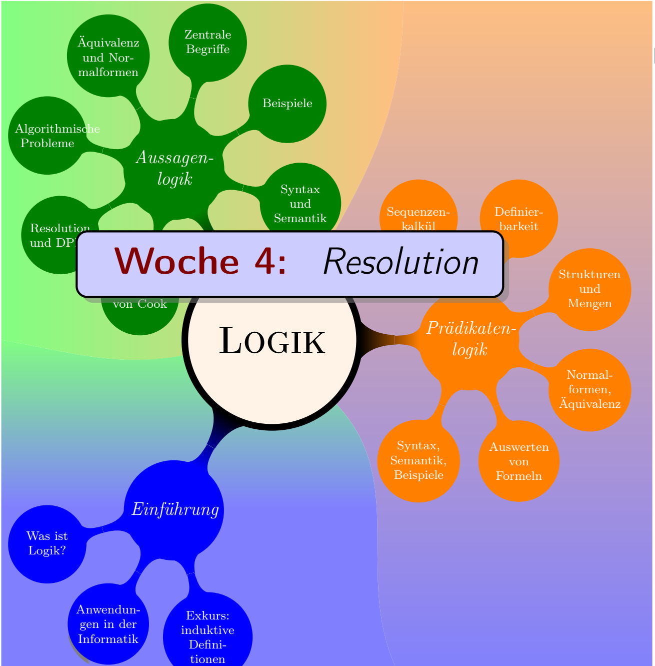
Logik im Wintersemester 2022/2023



Woche 4: *Resolution*

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*
	28	29	30	1	2	3	4	*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*

Wiederholung

Definition der wichtigsten Begriffe der Logik

Seien $\varphi, \psi \in \mathrm{AL}$ Formeln und $\Phi, \Psi \subseteq \mathrm{AL}$ Formelmengen.

Folgerung. ψ *folgt* aus φ, geschrieben $\varphi \models \psi$, wenn für alle zu φ und ψ passenden Belegungen β gilt: Wenn $\beta \models \varphi$, dann auch $\beta \models \psi$.

Äquivalenz. φ und ψ sind äquivalent, geschrieben $\varphi \equiv \psi$, wenn für alle zu φ, ψ passenden Belegungen β gilt: $\beta \models \varphi$ genau dann, wenn $\beta \models \psi$.

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

Erfüllbarkeit. φ ist erfüllbar, wenn es eine Belegung β gibt, die φ erfüllt.

Anderenfalls ist φ unerfüllbar.

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

Nützliche Äquivalenzen

Theorem. Für alle $\psi, \varphi, \vartheta \in \mathrm{AL}$:

1. $\neg\neg\varphi \equiv \varphi$ (Elimination doppelter Negation)
2. $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$ (Elimination der Implikation)
3. $\varphi \leftrightarrow \psi \equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$ (Elimination der Biimplikation)
4. $\neg(\psi \wedge \varphi) \equiv \neg\psi \vee \neg\varphi$ (de Morgansche Regeln)
   $\neg(\psi \vee \varphi) \equiv \neg\psi \wedge \neg\varphi$
5. $\psi \wedge (\varphi \vee \vartheta) \equiv (\psi \wedge \varphi) \vee (\psi \wedge \vartheta)$ (Distributivität)
   $\psi \vee (\varphi \wedge \vartheta) \equiv (\psi \vee \varphi) \wedge (\psi \vee \vartheta)$
6. $\psi \wedge (\psi \vee \varphi) \quad \equiv \quad \psi \vee (\psi \wedge \varphi) \quad \equiv \quad \psi$ (Absorbtionsgesetz)
7. $\psi \wedge \varphi \equiv \varphi \wedge \psi$ (Kommutativität von $\wedge$ und $\vee$)
   $\psi \vee \varphi \equiv \varphi \vee \psi$
8. $\psi \wedge (\varphi \wedge \vartheta) \equiv (\psi \wedge \varphi) \wedge \vartheta$ (Assoziativität von $\wedge$ und $\vee$)
   $\psi \vee (\varphi \vee \vartheta) \equiv (\psi \vee \varphi) \vee \vartheta$

Substitution

Definition. Für jede Formel $\varphi \in \mathsf{AL}$ und Substitution $\mathcal{S}$ definieren wir die Formel $\varphi\mathcal{S} \in \mathsf{AL}$ induktiv wie folgt:

Induktionsbasis.

- $\bot\mathcal{S} := \bot \qquad \top\mathcal{S} := \top$
- Wenn $X \in \mathsf{AVar}$, dann $X\mathcal{S} := \begin{cases} \mathcal{S}(X) & \text{wenn } X \in \mathrm{def}(\mathcal{S}) \\ X & \text{sonst} \end{cases}$

Induktionsschritt.

- $(\neg\varphi)\mathcal{S} := \neg(\varphi\mathcal{S})$
- Für $* \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ definieren wir

$$(\varphi * \psi)\mathcal{S} := (\varphi\mathcal{S} \; * \; \psi\mathcal{S}).$$

Informell. $\varphi\mathcal{S}$ entsteht aus φ indem alle Variablen $X \in \mathrm{def}(\mathcal{S})$ durch $\mathcal{S}(X)$ ersetzt werden.

Substitution. . partielle Abbildung $\mathcal{S} : \mathsf{AVar} \rightarrow \mathsf{AL}$ mit endlichem Definitionsbereich.

Beispiel. Sei $\varphi := V_1 \wedge V_2 \rightarrow V_1 \vee V_3$.

Dann gilt

$$\begin{aligned} \varphi\mathcal{S} &= (V_1 \wedge V_2)\mathcal{S} \rightarrow (V_1 \vee V_3)\mathcal{S} \\ &= V_1\mathcal{S} \wedge V_2\mathcal{S} \rightarrow V_1\mathcal{S} \vee V_3\mathcal{S} \\ &= V_2 \wedge (V_0 \vee V_1) \rightarrow V_2 \vee V_3. \end{aligned}$$

Das Substitutionslemma (formal)

Substitutionslemma.

Sei $\mathcal{S}$ eine Substitution und seien $\varphi, \varphi' \in \mathsf{AL}$ Formeln. Dann gilt

$$\varphi \equiv \varphi' \quad \Rightarrow \quad \varphi\mathcal{S} \equiv \varphi'\mathcal{S}.$$

Ersetzungslemma.

Sei $\varphi \in \mathsf{AL}$ eine Formel und ψ eine Unterformel von φ.

Sei φ' eine Formel, die man aus φ erhält, indem man ein Vorkommen der Unterformel ψ durch eine äquivalente Formel $\psi' \equiv \psi$ ersetzt.

Dann gilt $\varphi \equiv \varphi'$.

Boolesche Funktionen

Definition. Eine (n-stellige) Boolesche Funktion, ist eine Funktion

$$f : \{0,1\}^n \to \{0,1\}.$$

$\mathbb{B}^n$: Menge aller n-stelligen Booleschen Funktionen.

Proposition. Jede Formel $\varphi(X_1,\dots,X_n) \in \mathrm{AL}$ definiert eine Boolesche Funktion $f_\varphi := f(\varphi)$ mit

$$\begin{aligned} f_\varphi &: \{0,1\}^n \to \{0,1\} \\ f_\varphi(v_1,\dots,v_n) &:= [\![\varphi]\!]^\beta \end{aligned}$$

wobei $\beta(X_i) := v_i$, für alle $1 \leq i \leq n$.

Theorem. Zu jeder Booleschen Funktion $f : \{0,1\}^n \to \{0,1\}$ gibt es eine Formel $\varphi(X_1,\dots,X_n)$, so dass $f(\varphi) = f$.

Boolesche Funktionen

Definition. Eine (n-stellige) Boolesche Funktion, ist eine Funktion

$$f : \{0,1\}^n \to \{0,1\}.$$

$\mathbb{B}^n$: Menge aller n-stelligen Booleschen Funktionen.

Proposition. Jede Formel $\varphi(X_1, \dots, X_n) \in \mathrm{AL}$ definiert eine Boolesche Funktion $f_\varphi := f(\varphi)$ mit

$$\begin{aligned} f_\varphi \quad &: \quad \{0,1\}^n \to \{0,1\} \\ f_\varphi(v_1, \dots, v_n) \quad &:= \quad [\![\varphi]\!]^\beta \end{aligned}$$

wobei $\beta(X_i) := v_i$, für alle $1 \leq i \leq n$.

Theorem. Zu jeder Booleschen Funktion $f : \{0,1\}^n \to \{0,1\}$ gibt es eine Formel $\varphi(X_1, \dots, X_n)$, so dass $f(\varphi) = f$.

Boolesche Funktionen

Definition. Eine (n-stellige) Boolesche Funktion, ist eine Funktion

$$f : \{0,1\}^n \to \{0,1\}.$$

$\mathbb{B}^n$: Menge aller n-stelligen Booleschen Funktionen.

Proposition. Jede Formel $\varphi(X_1, \dots, X_n) \in \mathrm{AL}$ definiert eine Boolesche Funktion $f_\varphi := f(\varphi)$ mit

$$\begin{aligned} f_\varphi &: \{0,1\}^n \to \{0,1\} \\ f_\varphi(v_1, \dots, v_n) &:= [\![\varphi]\!]^\beta \end{aligned}$$

wobei $\beta(X_i) := v_i$, für alle $1 \leq i \leq n$.

Theorem. Zu jeder Booleschen Funktion $f : \{0,1\}^n \to \{0,1\}$ gibt es eine Formel $\varphi(X_1, \dots, X_n)$, so dass $f(\varphi) = f$.

Beispiel.
$\varphi := (X_1 \wedge X_2) \vee (X_1 \wedge X_3) \vee (X_2 \wedge X_3)$

$f_\varphi(1,0,1) = 1$	$f_\varphi(0,1,1) = 1$
$f_\varphi(1,1,0) = 1$	$f_\varphi(1,1,1) = 1$
$f_\varphi(v_1, v_2, v_3) = 0$	sonst

$f_\varphi(\bar{v})$: Mehrheitsfunktion

3.5 Normalformen

Normalformen

Normalformen. Eine Normalform der Aussagenlogik ist eine Klasse aussagenlogischer Formeln, so dass jede Formel in AL zu einer Formel in dieser Normalform äquivalent ist.

Reduzierte Formeln

Die schon bekannten Äquivalenzen liefern

$$\begin{aligned} \varphi \rightarrow \psi &\equiv \neg\varphi \vee \psi \\ \varphi \leftrightarrow \psi &\equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi) \equiv (\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi) \\ \varphi \wedge \psi &\equiv \neg(\neg\varphi \vee \neg\psi) \end{aligned}$$

Äquivalenzen.

1. $\neg\neg\varphi \equiv \varphi$
2. $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
3. $\varphi \leftrightarrow \psi \equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$
4. $\neg(\psi \wedge \varphi) \equiv \neg\psi \vee \neg\varphi$
   $\neg(\psi \vee \varphi) \equiv \neg\psi \wedge \neg\varphi$
5. $\psi \wedge (\varphi \vee \vartheta) \equiv (\psi \wedge \varphi) \vee (\psi \wedge \vartheta)$
   $\psi \vee (\varphi \wedge \vartheta) \equiv (\psi \vee \varphi) \wedge (\psi \vee \vartheta)$
6. $\psi \wedge (\psi \vee \varphi) \equiv \psi \vee (\psi \wedge \varphi) \equiv \psi$
7. $\psi \wedge \varphi \equiv \varphi \wedge \psi$
   $\psi \vee \varphi \equiv \varphi \vee \psi$
8. $\psi \wedge (\varphi \wedge \vartheta) \equiv (\psi \wedge \varphi) \wedge \vartheta$
   $\psi \vee (\varphi \vee \vartheta) \equiv (\psi \vee \varphi) \vee \vartheta$

Reduzierte Formeln

Die schon bekannten Äquivalenzen liefern

$$\begin{aligned}
\varphi \rightarrow \psi &\equiv \neg\varphi \vee \psi \\
\varphi \leftrightarrow \psi &\equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi) \equiv (\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi) \\
\varphi \wedge \psi &\equiv \neg(\neg\varphi \vee \neg\psi)
\end{aligned}$$

Korollar.

Jede aussagenlogische Formel ist äquivalent zu einer Formel ohne $\wedge$, $\rightarrow$, $\leftrightarrow$, d.h. in der nur $\top$, $\bot$, Variablen und $\vee$ und $\neg$ vorkommen.

Wir nennen solche Formeln reduziert.

Äquivalenzen.

1. $\neg\neg\varphi \equiv \varphi$
2. $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
3. $\varphi \leftrightarrow \psi \equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$
4. $\neg(\psi \wedge \varphi) \equiv \neg\psi \vee \neg\varphi$
   $\neg(\psi \vee \varphi) \equiv \neg\psi \wedge \neg\varphi$
5. $\psi \wedge (\varphi \vee \vartheta) \equiv (\psi \wedge \varphi) \vee (\psi \wedge \vartheta)$
   $\psi \vee (\varphi \wedge \vartheta) \equiv (\psi \vee \varphi) \wedge (\psi \vee \vartheta)$
6. $\psi \wedge (\psi \vee \varphi) \equiv \psi \vee (\psi \wedge \varphi) \equiv \psi$
7. $\psi \wedge \varphi \equiv \varphi \wedge \psi$
   $\psi \vee \varphi \equiv \varphi \vee \psi$
8. $\psi \wedge (\varphi \wedge \vartheta) \equiv (\psi \wedge \varphi) \wedge \vartheta$
   $\psi \vee (\varphi \vee \vartheta) \equiv (\psi \vee \varphi) \vee \vartheta$

Negationsnormalform

Normalform. Klasse aussagenlogischer Formeln, so dass jede Formel in AL zu einer Formel in dieser Normalform äquivalent ist.

$X \to Y$ $\neg X \vee Y$

Definition. Eine ~~Formel~~ $\varphi \in \mathrm{AL}$ ist in Negationsnormalform (NNF), wenn die Symbole $\rightarrow$ und $\leftrightarrow$ nicht vorkommen und Negation nur vor atomaren Formeln auftritt.

$\neg\bot \equiv \top$

Beispiel. $\varphi := (\neg X \vee Y) \wedge \neg Z$

Negationsnormalform

Normalform. Klasse aussagenlogischer Formeln, so dass jede Formel in AL zu einer Formel in dieser Normalform äquivalent ist.

Definition. Eine Formel $\varphi \in \mathrm{AL}$ ist in Negationsnormalform (NNF), wenn die Symbole $\rightarrow$ und $\leftrightarrow$ nicht vorkommen und Negation nur vor atomaren Formeln auftritt.

Beispiel. $\varphi := (\neg X \vee Y) \wedge \neg Z$

Theorem. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formel $\varphi^* \in \mathrm{AL}$ in Negationsnormalform.

Ein Algorithmus für die NNF

Eingabe. Eine Formel $\varphi \in \mathsf{AL}$ ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.

Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

NNF(φ). Eingabe. Eine Formel $\varphi \in \mathrm{AL}$ ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.
Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

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

NNF(φ). Eingabe. Eine Formel $\varphi \in \mathrm{AL}$ ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.
Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

$$
\begin{aligned}
& \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z)) \\
\equiv \quad & (\neg\neg(X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z))
\end{aligned}
$$

NNF(φ). Eingabe. Eine Formel $\varphi \in$ AL ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.

Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

$$
\begin{aligned}
& \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z)) \\
\equiv\ & (\neg\neg(X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z))
\end{aligned}
$$

NNF(φ). Eingabe. Eine Formel $\varphi \in$ AL ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.
Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

$$
\begin{aligned}
& \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z)) \\
\equiv \quad & (\neg\neg(X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv \quad & ((X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv \quad & ((X \vee Y) \vee \neg((\neg X \vee \neg Y) \vee Z))
\end{aligned}
$$

NNF(φ). Eingabe. Eine Formel $\varphi \in$ AL ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.

Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

$$
\begin{aligned}
& \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z)) \\
\equiv \quad & (\neg\neg(X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv \quad & ((X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv \quad & ((X \vee Y) \vee \neg((\neg X \vee \neg Y) \vee Z)) \\
\equiv \quad & ((X \vee Y) \vee (\neg(\neg X \vee \neg Y) \wedge \neg Z))
\end{aligned}
$$

NNF(φ). Eingabe. Eine Formel $\varphi \in$ AL ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.
Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

$$
\begin{aligned}
& \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z)) \\
\equiv\ & (\neg\neg(X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee \neg((\neg X \vee \neg Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee (\neg(\neg X \vee \neg Y) \wedge \neg Z)) \\
\equiv\ & ((X \vee Y) \vee ((\neg\neg X \wedge \neg\neg Y) \wedge \neg Z))
\end{aligned}
$$

NNF(φ). Eingabe. Eine Formel $\varphi \in$ AL ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.
Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

$$
\begin{aligned}
& \neg(\neg(X \vee Y) \wedge (\neg(X \wedge Y) \vee Z)) \\
\equiv\ & (\neg\neg(X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee \neg(\neg(X \wedge Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee \neg((\neg X \vee \neg Y) \vee Z)) \\
\equiv\ & ((X \vee Y) \vee (\neg(\neg X \vee \neg Y) \wedge \neg Z)) \\
\equiv\ & ((X \vee Y) \vee ((\neg\neg X \wedge \neg\neg Y) \wedge \neg Z)) \\
\equiv\ & ((X \vee Y) \vee ((X \wedge \neg\neg Y) \wedge \neg Z))
\end{aligned}
$$

NNF(φ). Eingabe. Eine Formel $\varphi \in \mathrm{AL}$ ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.

Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \wedge \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \vee \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \vee \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \wedge \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Beispiel

Sei $\varphi := \neg(\neg(X \lor Y) \land (\neg(X \land Y) \lor Z))$.

Der Algorithmus erzeugt folgende Zwischenformeln:

$$
\begin{aligned}
& \neg(\neg(X \lor Y) \land (\neg(X \land Y) \lor Z)) \\
\equiv \quad & (\neg\neg(X \lor Y) \lor \neg(\neg(X \land Y) \lor Z)) \\
\equiv \quad & ((X \lor Y) \lor \neg(\neg(X \land Y) \lor Z)) \\
\equiv \quad & ((X \lor Y) \lor \neg((\neg X \lor \neg Y) \lor Z)) \\
\equiv \quad & ((X \lor Y) \lor (\neg(\neg X \lor \neg Y) \land \neg Z)) \\
\equiv \quad & ((X \lor Y) \lor ((\neg\neg X \land \neg\neg Y) \land \neg Z)) \\
\equiv \quad & ((X \lor Y) \lor ((X \land \neg\neg Y) \land \neg Z)) \\
\equiv \quad & ((X \lor Y) \lor ((X \land Y) \land \neg Z))
\end{aligned}
$$

NNF(φ). Eingabe. Eine Formel $\varphi \in$ AL ohne $\rightarrow, \leftrightarrow$.

Ausgabe. Eine Formel $\varphi^* \equiv \varphi$ in NNF

Algorithmus. Wiederhole die folgenden Schritte.
Wenn φ in NNF ist, gib $\varphi^* := \varphi$ aus.

Wenn φ eine Unterformel ψ der Form $\neg\neg\psi_1$ enthält,
dann ersetze ψ durch ψ_1 um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \land \psi_2)$ enthält,
dann ersetze ψ durch $(\neg\psi_1 \lor \neg\psi_2)$ um φ' zu erhalten.

Wenn φ eine Unterformel ψ der Form $\neg(\psi_1 \lor \psi_2)$ enthält
dann ersetze ψ durch $(\neg\psi_1 \land \neg\psi_2)$ um φ' zu erhalten.

Setze $\varphi := \varphi'$.

Korrektheit und Vollständigkeit

Lemma. Der Algorithmus terminiert auf jeder gültigen Eingabe $\varphi \in \mathsf{AL}$ und konstruiert eine Formel φ^* in NNF, so dass $\varphi \equiv \varphi^*$.

Korrektheit und Vollständigkeit

Lemma. Der Algorithmus terminiert auf jeder gültigen Eingabe $\varphi \in \mathsf{AL}$ und konstruiert eine Formel φ^* in NNF, so dass $\varphi \equiv \varphi^*$.

Äquivalenz. Folgt aus dem Ersetzungslemma.

Korrektheit und Vollständigkeit

Lemma. Der Algorithmus terminiert auf jeder gültigen Eingabe $\varphi \in \mathsf{AL}$ und konstruiert eine Formel φ^* in NNF, so dass $\varphi \equiv \varphi^*$.

Äquivalenz. Folgt aus dem Ersetzungslemma.

Terminierung. Wir definieren eine Funktion

$$h : \mathsf{AL} \to \mathbb{N},$$

die die Höhe einer Formel angibt, wie folgt:

- Ist ψ atomar, so gilt $h(\psi) := 0$.
- Ist $\psi := \neg\psi'$, so gilt $h(\psi) := 1 + h(\psi')$.
- Ist $\psi := (\psi_1 \vee \psi_2)$ oder $\psi := (\psi_1 \wedge \psi_2)$, so gilt $h(\psi) := 1 + \max\{h(\psi_1), h(\psi_2)\}$.

Die Höhe einer Formel ist also die Höhe des Syntaxbaums der Formel.

Korrektheit und Vollständigkeit

Lemma. Der Algorithmus terminiert auf jeder gültigen Eingabe $\varphi \in \mathsf{AL}$ und konstruiert eine Formel φ^* in NNF, so dass $\varphi \equiv \varphi^*$.

Äquivalenz. Folgt aus dem Ersetzungslemma.

Terminierung. Wir definieren eine Funktion

$$h : \mathsf{AL} \to \mathbb{N},$$

die die Höhe einer Formel angibt, wie folgt:

- Ist ψ atomar, so gilt $h(\psi) := 0$.
- Ist $\psi := \neg\psi'$, so gilt $h(\psi) := 1 + h(\psi')$.
- Ist $\psi := (\psi_1 \vee \psi_2)$ oder $\psi := (\psi_1 \wedge \psi_2)$, so gilt $h(\psi) := 1 + \max\{h(\psi_1), h(\psi_2)\}$.

Die Höhe einer Formel ist also die Höhe des Syntaxbaums der Formel.

Beobachtung. Eine Formel ist in NNF genau dann, wenn jede Unterformel der Form $\neg\psi'$ die Höhe 1 hat.

Beweis des Lemmas

Sei jetzt $f(\varphi) := \sum\{3^{h(\psi)} : \psi = \neg\psi' \in \mathsf{sub}(\varphi)\}$

Beweis des Lemmas

Sei jetzt $f(\varphi) := \sum\{3^{h(\psi)} : \psi = \neg\psi' \in \mathsf{sub}(\varphi)\}$

Behauptung. Sei φ eine Formel und φ' die Formel, die aus φ in einem Schritt des Algorithmus' entsteht. Dann gilt $f(\varphi') < f(\varphi)$.

Beweis des Lemmas

$3 > 2$

Sei jetzt $f(\varphi) := \sum\{3^{h(\psi)} : \psi = \neg\psi' \in \mathrm{sub}(\varphi)\}$

Behauptung. Sei φ eine Formel und φ' die Formel, die aus φ in einem Schritt des Algorithmus' entsteht. Dann gilt $f(\varphi') < f(\varphi)$.

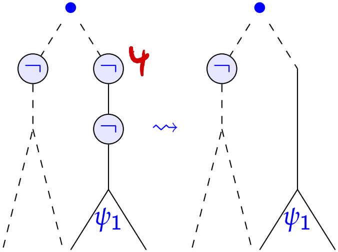
Beweis.

Angenommen, $\psi := \neg\neg\psi_1$.

Wir ersetzen also ψ durch ψ_1.

Dadurch erhöht sich die Tiefe der restlichen Negationsformeln nicht.

Die Zahl dieser Formeln reduziert sich um 2 und daher $f(\varphi') < f(\varphi)$.



Beweis des Lemmas

Sei jetzt $f(\varphi) := \sum\{3^{h(\psi)} : \psi = \neg\psi' \in \mathsf{sub}(\varphi)\}$.

Behauptung. Sei φ eine Formel und φ' die Formel, die aus φ in einem Schritt des Algorithmus' entsteht. Dann gilt $f(\varphi') < f(\varphi)$.

Beweis.

Angenommen, $\psi := \neg(\psi_1 \vee \psi_2)$ oder $\psi := \neg(\psi_1 \wedge \psi_2)$.

Beweis des Lemmas

$f(\varphi')$

Sei jetzt $f(\varphi) := \sum\{3^{h(\psi)} : \psi = \neg\psi' \in \mathrm{sub}(\varphi)\}$.

Behauptung. Sei φ eine Formel und φ' die Formel, die aus φ in einem Schritt des Algorithmus' entsteht. Dann gilt $f(\varphi') < f(\varphi)$.

Beweis.

Angenommen, $\psi := \neg(\psi_1 \vee \psi_2)$ oder $\psi := \neg(\psi_1 \wedge \psi_2)$.

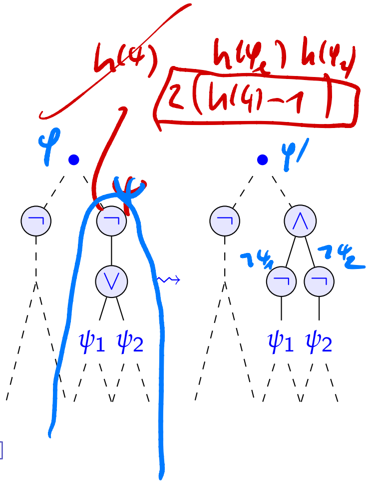
Dann bleibt die Tiefe aller Negationsformeln außer ψ gleich. $h(\psi)$

In der Summe wird $3^{h(\psi)}$ durch 2,0001

$$3^{h(\neg\psi_1)} + 3^{h(\neg\psi_2)} \le 2 \cdot 3^{h(\psi)-1} < 3^{h(\psi)}$$

ersetzt.

Also gilt $f(\varphi') < f(\varphi)$. □



Negationsnormalform

Normalform. Klasse aussagenlogischer Formeln, so dass jede Formel in AL zu einer Formel in dieser Normalform äquivalent ist.

Definition. Eine Formel $\varphi \in \mathsf{AL}$ ist in Negationsnormalform (NNF), wenn die Symbole $\rightarrow$ und $\leftrightarrow$ nicht vorkommen und Negation nur vor atomaren Formeln auftritt.

Beispiel. $\varphi := (\neg X \vee Y) \wedge \neg Z$

Negationsnormalform

Normalform. Klasse aussagenlogischer Formeln, so dass jede Formel in AL zu einer Formel in dieser Normalform äquivalent ist.

Definition. Eine Formel $\varphi \in \mathrm{AL}$ ist in Negationsnormalform (NNF), wenn die Symbole $\rightarrow$ und $\leftrightarrow$ nicht vorkommen und Negation nur vor atomaren Formeln auftritt.

Beispiel. $\varphi := (\neg X \vee Y) \wedge \neg Z$

Theorem. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formel $\varphi^* \in \mathrm{AL}$ in Negationsnormalform.

Normalformen

Definition. Ein Literal L ist eine Aussagenvariable $X \in \text{AVar}$ oder deren Negation $\neg X$.

Definiere $\quad \overline{L} := \begin{cases} \neg X & \text{if } L = X \\ X & \text{if } L = \neg X. \end{cases}$

Normalformen

Definition. Eine Formel $\varphi \in \mathrm{AL}$ ist in disjunktiver Normalform (DNF), wenn sie folgende Gestalt hat:

$$\bigvee_{i=1}^{n} \Big(\bigwedge_{j=1}^{n_i} L_{i,j} \Big) \qquad L_{i,j}: \text{ Literale}$$

φ ist in konjunktiver Normalform (KNF), wenn sie folgende Gestalt hat:

$$\bigwedge_{i=1}^{n} \Big(\bigvee_{j=1}^{n_i} L_{i,j} \Big) \qquad L_{i,j}: \text{ Literale}$$

Normalformen

Definition. Eine Formel $\varphi \in \mathrm{AL}$ ist in disjunktiver Normalform (DNF), wenn sie folgende Gestalt hat:

$$\bigvee_{i=1}^{n} \Big(\bigwedge_{j=1}^{n_i} L_{i,j} \Big) \qquad L_{i,j}: \text{ Literale}$$

φ ist in konjunktiver Normalform (KNF), wenn sie folgende Gestalt hat:

$$\bigwedge_{i=1}^{n} \Big(\bigvee_{j=1}^{n_i} L_{i,j} \Big) \qquad L_{i,j}: \text{ Literale}$$

Theorem.

1. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formel in disjunktiver Normalform.
2. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formel in konjunktiver Normalform.

Äquivalenzen.

1. $\neg\neg\varphi \equiv \varphi$
2. $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
3. $\varphi \leftrightarrow \psi \equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$
4. $\neg(\psi \wedge \varphi) \equiv \neg\psi \vee \neg\varphi$
   $\neg(\psi \vee \varphi) \equiv \neg\psi \wedge \neg\varphi$
5. $\psi \wedge (\varphi \vee \vartheta) \equiv (\psi \wedge \varphi) \vee (\psi \wedge \vartheta)$
   $\psi \vee (\varphi \wedge \vartheta) \equiv (\psi \vee \varphi) \wedge (\psi \vee \vartheta)$
6. $\psi \wedge (\psi \vee \varphi) \equiv \psi \vee (\psi \wedge \varphi) \equiv \psi$
7. $\psi \wedge \varphi \equiv \varphi \wedge \psi$
   $\psi \vee \varphi \equiv \varphi \vee \psi$
8. $\psi \wedge (\varphi \wedge \vartheta) \equiv (\psi \wedge \varphi) \wedge \vartheta$
   $\psi \vee (\varphi \vee \vartheta) \equiv (\psi \vee \varphi) \vee \vartheta$

Erinnerung: Boolesche Funktionen

Theorem. Zu jeder Booleschen Funktion $f : \{0,1\}^n \to \{0,1\}$ gibt es eine Formel $\varphi(X_1, \dots, X_n)$, so dass $f(\varphi) = f$.

Beweis. Für jede Sequenz $\overline{v} := (v_1, \dots, v_n) \in \{0,1\}^n$ definieren wir

$$\varphi_{\overline{v}} := \big(\textstyle\bigwedge_{v_i=1} X_i\big) \ \wedge \ \big(\textstyle\bigwedge_{v_i=0} \neg X_i\big)$$

Offensichtlich gilt für jede Belegung β, wenn $\beta \models \varphi_{\overline{v}}$ dann gilt für alle $1 \leq i \leq n$:

$$\beta(X_i) = 1 \iff v_i = 1.$$

Wir definieren nun die Funktion f durch die Formel

$$\varphi_f(X_1, \dots, X_n) := \bigvee_{\substack{\overline{v} \in \{0,1\}^n \\ f(\overline{v})=1}} \varphi_{\overline{v}}.$$ CDNF

Beispiel. $\varphi := (X_1 \wedge X_2) \vee (X_1 \wedge X_3) \vee (X_2 \wedge X_3)$

$$\begin{array}{ll} f_\varphi(1,0,1) = 1 & f_\varphi(0,1,1) = 1 \\ f_\varphi(1,1,0) = 1 & f_\varphi(1,1,1) = 1 \\ f_\varphi(v_1,v_2,v_3) = 0 & \text{sonst} \end{array}$$

$f_\varphi(\bar{v})$: Mehrheitsfunktion

Für $(v_1, v_2, v_3) = (0,1,1)$ ist

$$\varphi_{0,1,1} = (\neg X_1 \wedge X_2 \wedge X_3).$$

$$\begin{array}{lll} \varphi_f(X_1,X_2,X_3) := & & \\ & (\neg X_1 \wedge X_2 \wedge X_3) & (=: \varphi_{0,1,1}) \\ \vee & (X_1 \wedge \neg X_2 \wedge X_3) & (=: \varphi_{1,0,1}) \\ \vee & (X_1 \wedge X_2 \wedge \neg X_3) & (=: \varphi_{1,1,0}) \\ \vee & (X_1 \wedge X_2 \wedge X_3) & (=: \varphi_{1,1,1}) \end{array}$$

Normalformen

Theorem.

1. Jede Formel $\varphi \in \mathsf{AL}$ ist äquivalent zu einer Formel in disjunktiver Normalform.
2. Jede Formel $\varphi \in \mathsf{AL}$ ist äquivalent zu einer Formeln in konjunktiver Normalform.

Normalformen

Theorem.

1. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formel in disjunktiver Normalform.
2. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formeln in konjunktiver Normalform.

Beweis.

1. Teil 1 folgt aus dem Beweis der Äquivalenz zu Booleschen Funktionen, da die dort konstruierten Formeln in disjunktiver Normalform sind.

Normalformen

Theorem.

1. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formel in disjunktiver Normalform.
2. Jede Formel $\varphi \in \mathrm{AL}$ ist äquivalent zu einer Formeln in konjunktiver Normalform.

Beweis.

1. Teil 1 folgt aus dem Beweis der Äquivalenz zu Booleschen Funktionen, da die dort konstruierten Formeln in disjunktiver Normalform sind.
2. Sei $\varphi \in \mathrm{AL}$. $\varphi \equiv \neg\neg\varphi \equiv \neg \dots \bigvee \bigwedge L_{ij} \equiv \bigwedge \bigvee \overline{L_{ij}}$

   Nach Teil 1 ist $\neg\varphi$ äquivalent zu Formel $\psi := \bigvee_{i=1}^{n} \bigwedge_{j=1}^{n_i} L_{i,j}$ in DNF.

   Mit Hilfe der de Morganschen Gesetze erhält man

$$\varphi \equiv \neg \bigvee_{i=1}^{n} \bigwedge_{j=1}^{n_i} L_{i,j} \quad \equiv \quad \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n_i} \overline{L_{i,j}}. \qquad \square$$

Normalformen

Bemerkung.

- Formeln in disjunktiver Normalform können sehr effizient auf Erfüllbarkeit getestet werden.
- Allerdings gibt es Formeln $\varphi_n \in \mathrm{AL}$, für alle $n \in \mathbb{N}$, so dass die Länge der kürzesten zu φ_n äquivalenten Formeln in DNF exponentiell in der Länge von φ_n sind.
- Es gibt also im Allgemeinen keinen effizienten Weg um aussagenlogische Formeln in disjunktive oder konjunktive Normalform umzuwandeln.
- Jedoch kann zu jeder Formel $\varphi \in \mathrm{AL}$ in Polynomialzeit eine Formel $\psi \in \mathrm{AL}$ in konjunktiver Normalform konstruiert werden, so dass

  φ genau dann erfüllbar ist, wenn ψ erfüllbar ist.

  Dies wird in praktischen Anwendungen benutzt, da die meisten aktuellen SAT-Löser Formeln in KNF als Eingabe erwarten.

4.1 Algorithmische Logikprobleme

Algorithmische Logikprobleme

Algorithmische Probleme in der Logik.

Auswertungsproblem. Gegeben eine Belegung β und eine Formel $\varphi \in \text{AL}$, entscheide ob $\beta \models \varphi$.

Äquivalenzproblem. Gegeben $\varphi, \psi \in \text{AL}$, entscheide ob $\varphi \equiv \psi$.

Semantische Folgerung. Gegeben $\Phi \subseteq \text{AL}$ und $\psi \in \text{AL}$, entscheide ob $\Phi \models \psi$.

Erfüllbarkeitsproblem. Gegeben $\varphi \in \text{AL}$, entscheide, ob φ erfüllbar ist.
(Berechne ggf. eine erfüllende Belegung.)

Allgemeingültigkeitsproblem. Gegeben $\varphi \in \text{AL}$, entscheide, ob φ allgemeingültig ist.

Zentrale Begriffe

Folgerung $\varphi \models \psi$: für alle β gilt:
Wenn $\beta \models \varphi$, dann $\beta \models \psi$.

Äquivalenz $\varphi \equiv \psi$: für alle β gilt:
$\beta \models \varphi$ gdw. $\beta \models \psi$.

β erfüllt φ, ist Modell von φ,
kurz $\beta \models \varphi$, wenn $[\![\varphi]\!]^\beta = 1$.

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

φ allgemeingültig wenn jede passende Belegung φ erfüllt.

Algorithmische Logikprobleme

Algorithmische Probleme in der Logik.

Auswertungsproblem. Gegeben eine Belegung β und eine Formel $\varphi \in \mathrm{AL}$, entscheide ob $\beta \models \varphi$.

Äquivalenzproblem. Gegeben $\varphi, \psi \in \mathrm{AL}$, entscheide ob $\varphi \equiv \psi$.

Semantische Folgerung. Gegeben $\Phi \subseteq \mathrm{AL}$ und $\psi \in \mathrm{AL}$, entscheide ob $\Phi \models \psi$.

Erfüllbarkeitsproblem. Gegeben $\varphi \in \mathrm{AL}$, entscheide, ob φ erfüllbar ist.
(Berechne ggf. eine erfüllende Belegung.)

Allgemeingültigkeitsproblem. Gegeben $\varphi \in \mathrm{AL}$, entscheide, ob φ allgemeingültig ist.

Zeige. Diese Probleme können alle auf nur zwei Probleme reduziert werden: *Auswertungsproblem* und *Erfüllbarkeitsproblem*

Zentrale Begriffe

Folgerung $\varphi \models \psi$: für alle β gilt:
Wenn $\beta \models \varphi$, dann $\beta \models \psi$.

Äquivalenz $\varphi \equiv \psi$: für alle β gilt:
$\beta \models \varphi$ gdw. $\beta \models \psi$.

β erfüllt φ, ist Modell von φ, kurz $\beta \models \varphi$, wenn $[\![\varphi]\!]^\beta = 1$.

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

φ allgemeingültig wenn jede passende Belegung φ erfüllt.

Reduktion auf Erfüllbarkeit

Äquivalenzproblem. *Geben: Formeln $\varphi, \psi \in \mathsf{AL}$. Frage: Gilt $\varphi \equiv \psi$?*

Das Problem kann leicht auf semantische Folgerung reduziert werden.

Denn: $\varphi \equiv \psi$ gdw. $\varphi \models \psi$ und $\psi \models \varphi$.

Wenn wir Folgerungen entscheiden können, dann also auch Äquivalenz.

Zentrale Begriffe

Folgerung $\varphi \models \psi$: für alle β gilt:
Wenn $\beta \models \varphi$, dann $\beta \models \psi$.

Äquivalenz $\varphi \equiv \psi$: für alle β gilt:
$\beta \models \varphi$ gdw. $\beta \models \psi$.

β erfüllt φ, ist Modell von φ,
kurz $\beta \models \varphi$, wenn $[\![\varphi]\!]^\beta = 1$.

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

φ allgemeingültig wenn jede passende Belegung φ erfüllt.

Reduktion auf Erfüllbarkeit

Äquivalenzproblem. *Geben: Formeln* $\varphi, \psi \in \mathsf{AL}$. *Frage: Gilt* $\varphi \equiv \psi$*?*

Das Problem kann leicht auf semantische Folgerung reduziert werden.

Denn: $\varphi \equiv \psi$ gdw. $\varphi \models \psi$ und $\psi \models \varphi$.

Wenn wir Folgerungen entscheiden können, dann also auch Äquivalenz.

Folgerung. *Gegeben:* $\Phi \subseteq \mathsf{AL}$ *und Formel* $\psi \in \mathsf{AL}$. *Frage: Gilt* $\Phi \models \psi$*?*

Das Problem kann leicht auf Erfüllbarkeit reduziert werden.

Denn: $\Phi \models \psi$ gdw. $\Phi \cup \{\neg\psi\}$ unerfüllbar.

Wenn wir Erfüllbarkeit entscheiden können, dann also auch Folgerung.

Zentrale Begriffe

Folgerung $\varphi \models \psi$: für alle β gilt:
Wenn $\beta \models \varphi$, dann $\beta \models \psi$.

Äquivalenz $\varphi \equiv \psi$: für alle β gilt:
$\beta \models \varphi$ gdw. $\beta \models \psi$.

β erfüllt φ, ist Modell von φ,
kurz $\beta \models \varphi$, wenn $[\![\varphi]\!]^{\beta} = 1$.

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

φ allgemeingültig wenn jede passende Belegung φ erfüllt.

Reduktion auf Erfüllbarkeit

Äquivalenzproblem. *Geben: Formeln $\varphi, \psi \in \mathsf{AL}$. Frage: Gilt $\varphi \equiv \psi$?*

Das Problem kann leicht auf semantische Folgerung reduziert werden.

Denn: $\varphi \equiv \psi$ gdw. $\varphi \models \psi$ und $\psi \models \varphi$.

Wenn wir Folgerungen entscheiden können, dann also auch Äquivalenz.

Folgerung. *Gegeben: $\Phi \subseteq \mathsf{AL}$ und Formel $\psi \in \mathsf{AL}$. Frage: Gilt $\Phi \models \psi$?*

Das Problem kann leicht auf Erfüllbarkeit reduziert werden.

Denn: $\Phi \models \psi$ gdw. $\Phi \cup \{\neg\psi\}$ unerfüllbar.

Wenn wir Erfüllbarkeit entscheiden können, dann also auch Folgerung.

Allgemeingültigkeit. *Gegeben: $\varphi \in \mathsf{AL}$. Frage: Ist φ allgemeingültig?*

Das Problem kann leicht auf Erfüllbarkeit reduziert werden.

Denn φ allgemeingültig gdw. $\neg\varphi$ unerfüllbar.

Zentrale Begriffe

Folgerung $\varphi \models \psi$: für alle β gilt:
Wenn $\beta \models \varphi$, dann $\beta \models \psi$.

Äquivalenz $\varphi \equiv \psi$: für alle β gilt:
$\beta \models \varphi$ gdw. $\beta \models \psi$.

β erfüllt φ, ist Modell von φ,
kurz $\beta \models \varphi$, wenn $[\![\varphi]\!]^\beta = 1$.

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

φ allgemeingültig wenn jede passende Belegung φ erfüllt.

Algorithmische Logikprobleme

Wir brauchen also nur noch zwei algorithmische Probleme zu lösen.

Algorithmische Probleme in der Logik.

Auswertungsproblem.

Gegeben eine Belegung β und eine Formel $\varphi \in \mathsf{AL}$, entscheide ob $\beta \models \varphi$.

Lösung. Das Auswertungsproblem der Aussagenlogik kann sehr leicht gelöst werden, z.B. durch einen einfachen rekursiven Algorithmus entsprechend der Definition der Semantikfunktion $[\![\varphi]\!]^\beta$.

Algorithmische Logikprobleme

Wir brauchen also nur noch zwei algorithmische Probleme zu lösen.

Algorithmische Probleme in der Logik.

Auswertungsproblem.

Gegeben eine Belegung β und eine Formel $\varphi \in \mathsf{AL}$, entscheide ob $\beta \models \varphi$.

Lösung. Das Auswertungsproblem der Aussagenlogik kann sehr leicht gelöst werden, z.B. durch einen einfachen rekursiven Algorithmus entsprechend der Definition der Semantikfunktion $[\![\varphi]\!]^{\beta}$.

Erfüllbarkeitsproblem.

Gegeben $\varphi \in \mathsf{AL}$, entscheide, ob φ erfüllbar ist.

(Berechne ggf. eine erfüllende Belegung.)

Lösung. In der Theorie? Schwierig. In der Praxis? Machbar.

Erfüllbarkeitstest

Wir brauchen also schnelle Verfahren um Formeln auf (Un-)Erfüllbarkeit zu testen.

Verfahren zum Test auf (Un-)Erfüllbarkeit von AL-Formeln.

- Wahrheitstafeln
- **Resolution**
- **Der DPLL Algorithmus**
- **Der aussagenlogische Sequenzenkalkül**
- ....

Das Wahrheitstafelverfahren

Das Wahrheitstafelverfahren.

Eingabe: Eine Formel $\varphi \in \mathsf{AL}$.

Ziel: entscheide, ob φ erfüllbar ist.

Methode:
1. Berechne die Wahrheitstafel für φ.
2. Überprüfe, ob die letzte Spalte eine 1 enthält.

Bemerkung. Für Allgemeingültigkeit entscheide, ob die letzte Spalte nur 1 enthält.

Effizienz des Wahrheitstafelverfahrens

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

Das macht das Wahrheitstafelverfahren extrem ineffizient außer für sehr kleine Formeln.

Variablen	Zeilen
10	$1{,}024 \approx 10^3$
20	$1{,}048{,}576 \approx 10^6$
30	$1{,}073{,}741{,}824 \approx 10^9$
40	$1{,}099{,}511{,}627{,}776 \approx 10^{12}$
50	$1{,}125{,}899{,}906{,}842{,}624 \approx 10^{15}$
60	$1{,}152{,}921{,}504{,}606{,}846{,}976 \approx 10^{18}$

Das Aussagenlogische Erfüllbarkeitsproblem

Bemerkung.

- Das Erfüllbarkeitsproblem der Aussagenlogik ist eines der am besten studierten Probleme der Informatik.
- Es ist "schwer" zu lösen (NP-vollständig, werden wir später beweisen)
- Allerdings existieren Verfahren, die das Problem für viele in der Praxis vorkommende Formeln sehr effizient lösen können.
  (Das Wahrheitstafelverfahren gehört nicht dazu)
- Diese haben wichtige Anwendungen in der Informatik, z.B. in der Verifikation.

4.2 Einführung in die Resolution

Der Resolutionskalkül

Der Resolutionskalkül ist ein Verfahren um die Unerfüllbarkeit von Formeln in konjunktiver Normalform zu beweisen.

Der Kalkül enthält nur eine einzige Regel und lässt sich daher sehr einfach implementieren.

Wir werden uns zunächst auf endliche Formelmengen beschränken.

Danach werden wir einen Satz beweisen, der uns in bestimmten Situationen auch die Behandlung unendlicher Formelmengen erlaubt.

Aussagenlogische Resolution: Beispiel

Wir wollen zeigen, dass die folgende Formel φ unerfüllbar ist.

$$\neg V \wedge (Y \vee Z \vee V) \wedge (\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (\neg W \vee Z)$$

Angenommen, φ wäre erfüllbar. Sei β eine Belegung, so dass $\beta \models \varphi$.

Aussagenlogische Resolution: Beispiel

Wir wollen zeigen, dass die folgende Formel φ unerfüllbar ist.

$$\neg V \wedge (Y \vee Z \vee V) \wedge (\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (\neg W \vee Z)$$

Angenommen, φ wäre erfüllbar. Sei β eine Belegung, so dass $\beta \models \varphi$.

- Offensichtlich gilt, $\beta \models \neg V$

Aussagenlogische Resolution: Beispiel

Wir wollen zeigen, dass die folgende Formel φ unerfüllbar ist.

$$\neg V \wedge (Y \vee Z \vee V) \wedge (\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (\neg W \vee Z)$$

Angenommen, φ wäre erfüllbar. Sei β eine Belegung, so dass $\beta \models \varphi$.

- Offensichtlich gilt, $\beta \models \neg V$
- Aus $\beta \models \neg V$ und $\beta \models Y \vee Z \vee V$ folgt $\beta \models Y \vee Z$

Aussagenlogische Resolution: Beispiel

Wir wollen zeigen, dass die folgende Formel φ unerfüllbar ist.

$$\neg V \wedge (Y \vee Z \vee V) \wedge (\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (\neg W \vee Z)$$

Angenommen, φ wäre erfüllbar. Sei β eine Belegung, so dass $\beta \models \varphi$.

- Offensichtlich gilt, $\beta \models \neg V$
- Aus $\beta \models \neg V$ und $\beta \models Y \vee Z \vee V$ folgt $\beta \models Y \vee Z$
- Aus $\beta \models Y \vee Z$ und $\beta \models \neg Y \vee W$ folgt $\beta \models Z \vee W$

Aussagenlogische Resolution: Beispiel

Wir wollen zeigen, dass die folgende Formel φ unerfüllbar ist.

$$\neg V \wedge (Y \vee Z \vee V) \wedge (\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (\neg W \vee Z)$$

Angenommen, φ wäre erfüllbar. Sei β eine Belegung, so dass $\beta \models \varphi$.

- Offensichtlich gilt, $\beta \models \neg V$
- Aus $\beta \models \neg V$ und $\beta \models Y \vee Z \vee V$ folgt $\beta \models Y \vee Z$
- Aus $\beta \models Y \vee Z$ und $\beta \models \neg Y \vee W$ folgt $\beta \models Z \vee W$
- Aus $\beta \models Z \vee W$ und $\beta \models \neg W \vee Z$ folgt $\beta \models Z$

Aussagenlogische Resolution: Beispiel

Wir wollen zeigen, dass die folgende Formel φ unerfüllbar ist.

$$\neg V \wedge (Y \vee Z \vee V) \wedge (\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (\neg W \vee Z)$$

Angenommen, φ wäre erfüllbar. Sei β eine Belegung, so dass $\beta \models \varphi$.

- Offensichtlich gilt, $\beta \models \neg V$
- Aus $\beta \models \neg V$ und $\beta \models Y \vee Z \vee V$ folgt $\beta \models Y \vee Z$
- Aus $\beta \models Y \vee Z$ und $\beta \models \neg Y \vee W$ folgt $\beta \models Z \vee W$
- Aus $\beta \models Z \vee W$ und $\beta \models \neg W \vee Z$ folgt $\beta \models Z$
- Aus $\beta \models \neg X \vee \neg Z$ und $\beta \models X \vee \neg Z$ folgt aber auch $\beta \models \neg Z$

Aussagenlogische Resolution: Beispiel

$(\neg X \vee Z) \wedge (X \vee Z) \equiv Z$

Wir wollen zeigen, dass die folgende Formel φ unerfüllbar ist.

$$\neg V \wedge (Y \vee Z \vee V) \wedge (\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (\neg W \vee Z)$$

Angenommen, φ wäre erfüllbar. Sei β eine Belegung, so dass $\beta \models \varphi$.

$\bigwedge \left(\bigvee L_{i,j} \right) \wedge \bigwedge (V)$

- Offensichtlich gilt, $\beta \models \neg V$
- Aus $\beta \models \neg V$ und $\beta \models Y \vee Z \vee V$ folgt $\beta \models Y \vee Z$
- Aus $\beta \models Y \vee Z$ und $\beta \models \neg Y \vee W$ folgt $\beta \models Z \vee W$
- Aus $\beta \models Z \vee W$ und $\beta \models \neg W \vee Z$ folgt $\beta \models Z$
- Aus $\beta \models \neg X \vee \neg Z$ und $\beta \models X \vee \neg Z$ folgt aber auch $\beta \models \neg Z$
- Offensichtlich ist das ein Widerspruch zu $\beta \models Z$.

$\square$

Aussagenlogische Resolution

$$\underbrace{(X \vee Y)}_{F_1} \wedge \underbrace{(X \vee Z)}_{F_2}$$

Die aussagenlogische Resolution ist eine Methode um zu zeigen, dass eine Formel in konjunktiver Normalform nicht erfüllbar ist.

Theorem. Zu jeder Formel φ gibt es eine Formel ψ in KNF, so dass

1. φ ist genau dann erfüllbar, wenn ψ erfüllbar ist.
2. $|\psi| \leq c \cdot |\varphi|$ für eine Konstante $c \in \mathbb{N}$ unabhängig von φ.
3. ψ kann aus φ effizient (in Linearzeit) berechnet werden.

Notation

Eine Formel

$$(\neg X \vee \neg Z) \wedge (X \vee \neg Z) \wedge (\neg Y \vee W) \wedge (Y \vee Z \vee V) \wedge \neg V \wedge (\neg W \vee Z)$$

in KNF schreiben wir als Klauselmenge wie folgt:

Klausel

$$\{\neg X, \neg Z\}, \quad \{X, \neg Z\}, \quad \{\neg Y, W\}, \quad \{Y, Z, V\}, \quad \{\neg V\}, \quad \{\neg W, Z\}$$

D.h., $\quad Y \vee Z \vee V \quad \rightsquigarrow \quad \{Y, Z, V\}$.

Klauseln

Definition.

- Eine Klausel ist eine endliche Menge von Literalen.
- Zu $\varphi := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{i,j}$ definieren wir Menge $\mathcal{C}(\varphi)$ von Klauseln:
  - zu $\bigvee_{j=1}^{m_i} L_{i,j}$ definieren wir $C_i := \{L_{i,j} : 1 \leq j \leq m_i\}$
  - $\mathcal{C}(\varphi) := \{C_1, \dots, C_n\}$.

Klauseln

Definition.

- Eine Klausel ist eine endliche Menge von Literalen.
- Zu $\varphi := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{i,j}$ definieren wir Menge $\mathcal{C}(\varphi)$ von Klauseln:
  - zu $\bigvee_{j=1}^{m_i} L_{i,j}$ definieren wir $C_i := \{L_{i,j} : 1 \leq j \leq m_i\}$
  - $\mathcal{C}(\varphi) := \{C_1, \dots, C_n\}$.
- Umgekehrt, entspricht jeder Menge $\mathcal{C} := \{C_1, \dots, C_n\}$ von Klauseln

$$C_i := \{L_{i,j} : 1 \leq j \leq m_i\}$$

die Formel $\varphi(\mathcal{C}) := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{i,j}$. $\varphi(c_i)$ $\qquad \bigwedge \varphi(c_i)$

Falls $\mathcal{C} := \emptyset$, definieren wir $\varphi(\mathcal{C}) := \top$.

Beispiel.
$\varphi := (X \vee Y) \wedge (\neg X \vee Z)$

$C_1 := \{X, Y\}$
$C_2 := \{\neg X, Z\}$
$\mathcal{C}(\varphi) := \{C_1, C_2\}$.

Klauseln

Definition.

- Eine Klausel ist eine endliche Menge von Literalen.
- Zu $\varphi := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{i,j}$ definieren wir Menge $\mathcal{C}(\varphi)$ von Klauseln:
  - zu $\bigvee_{j=1}^{m_i} L_{i,j}$ definieren wir $C_i := \{L_{i,j} : 1 \le j \le m_i\}$
  - $\mathcal{C}(\varphi) := \{C_1, \dots, C_n\}$.
- Umgekehrt, entspricht jeder Menge $\mathcal{C} := \{C_1, \dots, C_n\}$ von Klauseln

$$C_i := \{L_{i,j} : 1 \le j \le m_i\}$$

die Formel $\varphi(\mathcal{C}) := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{i,j}$.

$\bigwedge_{C \in \mathcal{C}(\varphi)} \varphi(C) \wedge \top$

Falls $\mathcal{C} := \emptyset$, definieren wir $\varphi(\mathcal{C}) := \top$.

- Die leere Klausel wird mit $\Box$ bezeichnet und wir definieren $\varphi(\Box) := \bot$.

C_i $\varphi(C_i) = \bigvee_{L \in C_i} L$ $\bot$

Beispiel.
$\varphi := (X \vee Y) \wedge (\neg X \vee Z)$

$C_1 := \{X, Y\}$
$C_2 := \{\neg X, Z\}$
$\mathcal{C}(\varphi) := \{C_1, C_2\}$.

Formeln vs. Klauselmengen

Notation. Wir erweitern Notation für Formeln auf Klauselmengen.

- Für eine Belegung β und Klauselmenge $\mathcal{C}$ schreiben wir $\beta \models \mathcal{C}$ für $\beta \models \varphi(\mathcal{C})$
- Wir schreiben $\mathcal{C} \models C$ falls jede $\mathcal{C}$ erfüllende Belegung auch C erfüllt.
- Eine Klauselmenge $\mathcal{C}$ ist erfüllbar, wenn $\varphi(\mathcal{C})$ erfüllbar ist.
- Wenn $\mathcal{C}$ nur eine Klausel C enthält, schreiben wir einfach nur $\beta \models C$ etc.

Formeln vs. Klauselmengen

Notation. Wir erweitern Notation für Formeln auf Klauselmengen.

- Für eine Belegung β und Klauselmenge $\mathcal{C}$ schreiben wir $\beta \models \mathcal{C}$ für $\beta \models \varphi(\mathcal{C})$
- Wir schreiben $\mathcal{C} \models C$ falls jede $\mathcal{C}$ erfüllende Belegung auch C erfüllt.
- Eine Klauselmenge $\mathcal{C}$ ist erfüllbar, wenn $\varphi(\mathcal{C})$ erfüllbar ist.
- Wenn $\mathcal{C}$ nur eine Klausel C enthält, schreiben wir einfach nur $\beta \models C$ etc.

Beobachtung. Eine Belegung β erfüllt eine Klauselmenge $\mathcal{C}$, wenn jede Klausel $C \in \mathcal{C}$ ein Literal L enthält, so dass $[\![L]\!]^\beta = 1$.

Insbesondere ist also jede Klauselmenge, die die leere Klausel enthält, unerfüllbar.

4.3 Resolution

Resolution

Definition. Seien C, C_1, C_2 Klauseln.

C ist eine Resolvente von C_1, C_2, wenn es ein Literal L gibt mit

$$L \in C_1 \text{ und } \overline{L} \in C_2 \text{ und } C = (C_1 \setminus \{L\}) \cup (C_2 \setminus \{\overline{L}\}).$$

$$C = (C_1 \cup C_2) \setminus \{L, \overline{L}\}$$

Wir sagen, dass C_1 und C_2 resolviert werden. Die Menge der Resolventen von C_1 und C_2 bezeichnen wir mit $Res(C_1, C_2)$.

Graphische Darstellung

$$\begin{tikzcd}
C_1 \arrow[dr] & & C_2 \arrow[dl] \\
& C &
\end{tikzcd}$$

Erinnerung.

Für ein Literal L bezeichnet $\bar{L}$ das duale Literal, d.h. $\bar{X} = \neg X$ und $\overline{\neg X} = X$.

Klausel $C := \{L_1, \dots, L_n\}$ entspricht $\varphi(C) := \bigvee_{i=1}^{n} L_i$

Klauselmenge $\mathcal{C} := \{C_1, \dots, C_n\}$ entspricht $\varphi(\mathcal{C}) := \bigwedge_{i=1}^{n} \varphi(C_i)$.

Bemerkung

Zwei Klauseln können mehr als eine Resolvente haben.

Beispiel. Für $C_1 := \{X, Y, Z\}$ und $C_2 := \{\neg X, \neg Y, W\}$ gilt:

Resolvente 1

$$\begin{array}{ccc} \{\cancel{X}, Y, Z\} & & \{\cancel{\neg X}, \neg Y, W\} \\ & \{Y, \neg Y, Z, W\} & \end{array}$$

$Y \vee \neg Y \vee Z \vee W$

Resolvente 2

$$\begin{array}{ccc} \{X, \cancel{Y}, Z\} & & \{\neg X, \cancel{\neg Y}, W\} \\ & \{X, \neg X, Z, W\} & \end{array}$$

Dies ist aber ein degenerierter Fall, der im weiteren keine Rolle spielen wird.

Insbesondere ist in diesem Fall die Resolvente immer allgemeingültig, da sie ein Literal und sein duales Literal enthält.

> **Resolvente.**
> C_1, C_2 Klauseln,
>
> L Literal mit $L \in C_1$, $\overline{L} \in C_2$.
>
> Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Aussagenlogische Resolution

Lemma.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in Res(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.

Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Aussagenlogische Resolution

Lemma.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in \mathit{Res}(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis. Wir zeigen zunächst, dass $\{C_1, C_2\} \models C$.

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.

Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Aussagenlogische Resolution

Lemma.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in \mathit{Res}(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis. Wir zeigen zunächst, dass $\{C_1, C_2\} \models C$.

Zu zeigen: Jede Belegung β mit $\beta \models \{C_1, C_2\}$ erfüllt auch C.

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.

Resolvente $C =$
$C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Aussagenlogische Resolution

Lemma.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in \mathit{Res}(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis. Wir zeigen zunächst, dass $\{C_1, C_2\} \models C$.

Zu zeigen: Jede Belegung β mit $\beta \models \{C_1, C_2\}$ erfüllt auch C.

Sei β eine Belegung mit $\beta \models \{C_1, C_2\}$ und L das resolvierte Literal.

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.

Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Aussagenlogische Resolution

Lemma.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in Res(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis. Wir zeigen zunächst, dass $\{C_1, C_2\} \models C$.

Zu zeigen: Jede Belegung β mit $\beta \models \{\underset{L}{C_1}, \underset{\bar{L}}{C_2}\}$ erfüllt auch C.

Sei β eine Belegung mit $\beta \models \{C_1, C_2\}$ und L das resolvierte Literal.

Fall 1: $[\![L]\!]^\beta = 1$	**Fall 2:** $[\![L]\!]^\beta = 0$
Es gilt $\beta \models C_2$.	Es gilt $\beta \models C_1$.
Also existiert $L' \in C_2 \setminus \{\bar{L}\}$ mit $[\![L']\!]^\beta = 1$.	Also existiert $L' \in C_1 \setminus \{L\}$ mit $[\![L']\!]^\beta = 1$.
Da nach Definition $L' \in C$, folgt $\beta \models C$.	Da nach Definition $L' \in C$, folgt $\beta \models C$.

In beiden Fällen gilt also $\beta \models C$.

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\bar{L} \in C_2$.

Resolvente $C =$
$C_1 \setminus \{L\} \cup C_2 \setminus \{\bar{L}\}$.

Aussagenlogische Resolution

Lemma.
Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in \mathit{Res}(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis (Forts.). Wir zeigen nun, dass $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ äquivalent sind.

Aussagenlogische Resolution

Lemma.
Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in \mathit{Res}(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis (Forts.). Wir zeigen nun, dass $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ äquivalent sind.

Dazu müssen wir zeigen, dass für alle Belegungen β gilt:

$$\beta \models \mathcal{C} \text{ gdw. } \beta \models \mathcal{C} \cup \{C\}.$$

Aussagenlogische Resolution

Lemma.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in Res(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis (Forts.). Wir zeigen nun, dass $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ äquivalent sind.

Dazu müssen wir zeigen, dass für alle Belegungen β gilt:

$$\beta \models \mathcal{C} \text{ gdw. } \beta \models \mathcal{C} \cup \{C\}.$$

$\Leftarrow$: Wenn $\beta \models \mathcal{C} \cup \{C\}$, dann $\beta \models \mathcal{C}$.

Aussagenlogische Resolution

Lemma.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in Res(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis (Forts.). Wir zeigen nun, dass $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ äquivalent sind.

Dazu müssen wir zeigen, dass für alle Belegungen β gilt:

$$\beta \models \mathcal{C} \text{ gdw. } \beta \models \mathcal{C} \cup \{C\}.$$

$\Leftarrow$: Wenn $\beta \models \mathcal{C} \cup \{C\}$, dann $\beta \models \mathcal{C}$.

$\Rightarrow$: Bereits gesehen: $\{C_1, C_2\} \models C$. Da $C_1, C_2 \in \mathcal{C}$, folgt $\mathcal{C} \models \mathcal{C} \cup \{C\}$.

Aussagenlogische Resolution

Lemma.

Sei $\mathcal{C}$ eine Klauselmenge. Seien $C_1, C_2 \in \mathcal{C}$ und $C \in \mathit{Res}(C_1, C_2)$.

Dann gilt $\{C_1, C_2\} \models C$ und $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ sind äquivalent.

Beweis (Forts.). Wir zeigen nun, dass $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ äquivalent sind.

Dazu müssen wir zeigen, dass für alle Belegungen β gilt:

$$\beta \models \mathcal{C} \text{ gdw. } \beta \models \mathcal{C} \cup \{C\}.$$

$\Leftarrow$: Wenn $\beta \models \mathcal{C} \cup \{C\}$, dann $\beta \models \mathcal{C}$.

$\Rightarrow$: Bereits gesehen: $\{C_1, C_2\} \models C$. Da $C_1, C_2 \in \mathcal{C}$, folgt $\mathcal{C} \models \mathcal{C} \cup \{C\}$.

Also sind $\mathcal{C}$ und $\mathcal{C} \cup \{C\}$ äquivalent. $\square$

Resolutionsableitungen

Definition.

1. Eine Resolutionsableitung einer Klausel C aus einer Klauselmenge $\mathcal{C}$ ist eine Sequenz $(C_1, \dots, C_n)$, so dass
   - $C_n = C$ und
   - für alle $1 \leq i \leq n$ gilt

     $$C_i \in \mathcal{C} \quad \text{oder} \quad \text{es gibt } j, k < i \text{ mit } C_i \in Res(C_j, C_k).$$

   Wir sagen, dass C einen Resolutionsbeweis aus $\mathcal{C}$ hat und schreiben dies als $\mathcal{C} \vdash_R C$. $\models$

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.

Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Resolutionsableitungen

Definition.

1. Eine Resolutionsableitung einer Klausel C aus einer Klauselmenge $\mathcal{C}$ ist eine Sequenz $(C_1, \dots, C_n)$, so dass

   - $C_n = C$ und
   - für alle $1 \leq i \leq n$ gilt

   $$C_i \in \mathcal{C} \quad \text{oder} \quad \text{es gibt } j, k < i \text{ mit } C_i \in Res(C_j, C_k).$$

   Wir sagen, dass C einen Resolutionsbeweis aus $\mathcal{C}$ hat und schreiben dies als $\mathcal{C} \vdash_R C$.

2. Eine Resolutionswiderlegung einer Klauselmenge $\mathcal{C}$ ist eine Resolutionsableitung der leeren Klausel $\Box$.

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.

Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Beispiel einer Resolutionswiderlegung

$$\varphi := (V_1 \vee V_2) \wedge (\neg V_2 \vee V_n) \wedge (V_2 \vee \neg V_1) \wedge (\neg V_n \vee \neg V_4) \wedge (\neg V_n \vee V_4 \vee \neg V_5) \wedge (V_4 \vee V_5)$$

Klauselform.

$$\{V_1, V_2\} \quad \{\neg V_2, V_n\} \quad \{V_2, \neg V_1\} \quad \{\neg V_n, \neg V_4\} \quad \{\neg V_n, V_4, \neg V_5\} \quad \{V_4, V_5\}$$

Res.wid. von $\mathcal{C}$.
Sequenz $(C_1, \dots, C_n)$:
$C_n = C$ und
für alle $1 \leq i \leq n$ gilt
- $C_i \in \mathcal{C}$ oder
- es gibt $j, k < i$ mit $C_i \in Res(C_j, C_k)$.

Resolutionsableitung von $\Box$.

$$\Big(\{V_1, V_2\}, \quad \{\neg V_2, V_n\}, \quad \{V_1, V_n\}, \quad \{V_2, \neg V_1\}, \quad \{V_n, \neg V_1\}, \quad \{V_n\},$$
$$\{\neg V_n, \neg V_4\}, \quad \{\neg V_n, V_4, \neg V_5\}, \quad \{\neg V_n, V_4\}, \{V_4, V_5\}, \quad \{\neg V_n\}, \quad \Box\Big)$$

Resolvente.
C_1, C_2 Klauseln,
L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.
Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Beispiel einer Resolutionswiderlegung

$$\varphi := (V_1 \vee V_2) \wedge (\neg V_2 \vee V_n) \wedge (V_2 \vee \neg V_1) \wedge (\neg V_n \vee \neg V_4) \wedge (\neg V_n \vee V_4 \vee \neg V_5) \wedge (V_4 \vee V_5)$$

Klauselform.

$$\{V_1, \cancel{V_2}\} \quad \{\cancel{\neg V_2}, V_n\} \quad \{V_2, \neg V_1\} \quad \{\neg V_n, \neg V_4\} \quad \{\neg V_n, V_4, \neg V_5\} \quad \{V_4, V_5\}$$

$$\{V_1, V_n\}$$

Res.wid. von $\mathcal{C}$.
Sequenz $(C_1, \ldots, C_n)$:
$C_n = C$ und
für alle $1 \leq i \leq n$ gilt
- $C_i \in \mathcal{C}$ oder
- es gibt $j, k < i$ mit $C_i \in Res(C_j, C_k)$.

Resolutionsableitung von $\square$.

$$\big(\underset{C_1}{\{V_1, V_2\}}, \ \underset{C_2}{\{\neg V_2, V_n\}}, \ \underset{C_3}{\{V_1, V_n\}}, \ \{V_2, \neg V_1\}, \ \{V_n, \neg V_1\}, \ \{V_n\},$$
$$\{\neg V_n, \neg V_4\}, \ \{\neg V_n, V_4, \neg V_5\}, \ \{\neg V_n, V_4\}, \{V_4, V_5\}, \ \{\neg V_n\}, \ \square\big)$$

Resolvente.
C_1, C_2 Klauseln,
L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.
Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Beispiel einer Resolutionswiderlegung

$$\varphi := (V_1 \vee V_2) \wedge (\neg V_2 \vee V_n) \wedge (V_2 \vee \neg V_1) \wedge (\neg V_n \vee \neg V_4) \wedge (\neg V_n \vee V_4 \vee \neg V_5) \wedge (V_4 \vee V_5)$$

Klauselform.

$$\{V_1, V_2\} \quad \{\neg V_2, V_n\} \quad \{V_2, \neg V_1\} \quad \{\neg V_n, \neg V_4\} \quad \{\neg V_n, V_4, \neg V_5\} \quad \{V_4, V_5\}$$

$$\{V_1, V_n\} \qquad \{V_n, \neg V_1\}$$

Resolutionsableitung von $\Box$.

$$\Big(\{V_1, V_2\}, \quad \{\neg V_2, V_n\}, \quad \{V_1, V_n\}, \quad \{V_2, \neg V_1\}, \quad \{V_n, \neg V_1\}, \quad \{V_n\},$$
$$\{\neg V_n, \neg V_4\}, \quad \{\neg V_n, V_4, \neg V_5\}, \quad \{\neg V_n, V_4\}, \{V_4, V_5\}, \quad \{\neg V_n\}, \quad \Box\Big)$$

Res.wid. von $\mathcal{C}$.
Sequenz $(C_1, \dots, C_n)$:
$C_n = C$ und
für alle $1 \leq i \leq n$ gilt
- $C_i \in \mathcal{C}$ oder
- es gibt $j, k < i$ mit $C_i \in Res(C_j, C_k)$.

Resolvente.
C_1, C_2 Klauseln,
L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.
Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Beispiel einer Resolutionswiderlegung

$$\varphi := (V_1 \lor V_2) \land (\neg V_2 \lor V_n) \land (V_2 \lor \neg V_1) \land (\neg V_n \lor \neg V_4) \land (\neg V_n \lor V_4 \lor \neg V_5) \land (V_4 \lor V_5)$$

Klauselform.

$$\{V_1, V_2\} \quad \{\neg V_2, V_n\} \quad \{V_2, \neg V_1\} \quad \{\neg V_n, \neg V_4\} \quad \{\neg V_n, V_4, \neg V_5\} \quad \{V_4, V_5\}$$

$$\{V_1, V_n\} \quad \{V_n, \neg V_1\} \quad \{\neg V_n, V_4\}$$

Res.wid. von $\mathcal{C}$.
Sequenz $(C_1, \dots, C_n)$:
$C_n = C$ und
für alle $1 \leq i \leq n$ gilt
$C_i \in \mathcal{C}$ oder
es gibt $j, k < i$ mit
$C_i \in Res(C_j, C_k)$.

Resolutionsableitung von $\Box$.

$$\Big(\{V_1, V_2\}, \ \{\neg V_2, V_n\}, \ \{V_1, V_n\}, \ \{V_2, \neg V_1\}, \ \{V_n, \neg V_1\}, \ \{V_n\}, \ \{\neg V_n, \neg V_4\}, \ \{\neg V_n, V_4, \neg V_5\}, \ \{\neg V_n, V_4\}, \{V_4, V_5\}, \ \{\neg V_n\}, \ \Box\Big)$$

Resolvente.
C_1, C_2 Klauseln,
L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.
Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Beispiel einer Resolutionswiderlegung

$$\varphi := (V_1 \vee V_2) \wedge (\neg V_2 \vee V_n) \wedge (V_2 \vee \neg V_1) \wedge (\neg V_n \vee \neg V_4) \wedge (\neg V_n \vee V_4 \vee \neg V_5) \wedge (V_4 \vee V_5)$$

Klauselform.

$$\begin{tikzcd}[column sep=tiny]
\{V_1, V_2\} \arrow[dr] & & \{\neg V_2, V_n\} \arrow[dl] \arrow[dr] & & \{V_2, \neg V_1\} \arrow[dl] & \{\neg V_n, \neg V_4\} & \{\neg V_n, V_4, \neg V_5\} \arrow[dr] & & \{V_4, V_5\} \arrow[dl] \\
& \{V_1, V_n\} \arrow[dr] & & \{V_n, \neg V_1\} \arrow[dl] & & & & \{\neg V_n, V_4\} & \\
& & \{V_n\} & & & & & &
\end{tikzcd}$$

Res.wid. von $\mathcal{C}$.
Sequenz $(C_1, \ldots, C_n)$:
$C_n = C$ und
für alle $1 \leq i \leq n$ gilt
- $C_i \in \mathcal{C}$ oder
- es gibt $j, k < i$ mit $C_i \in Res(C_j, C_k)$.

Resolutionsableitung von $\Box$.

$$\big(\{V_1, V_2\},\ \{\neg V_2, V_n\},\ \{V_1, V_n\},\ \{V_2, \neg V_1\},\ \{V_n, \neg V_1\},\ \{V_n\},$$
$$\{\neg V_n, \neg V_4\},\ \{\neg V_n, V_4, \neg V_5\},\ \{\neg V_n, V_4\},\ \{V_4, V_5\},\ \{\neg V_n\},\ \Box\big)$$

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.

Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Beispiel einer Resolutionswiderlegung

$$\varphi := (V_1 \vee V_2) \wedge (\neg V_2 \vee V_n) \wedge (V_2 \vee \neg V_1) \wedge (\neg V_n \vee \neg V_4) \wedge (\neg V_n \vee V_4 \vee \neg V_5) \wedge (V_4 \vee V_5)$$

Klauselform.

$$\begin{tikzcd}
\{V_1, V_2\} \arrow[dr] & & \{\neg V_2, V_n\} \arrow[dl] \arrow[dr] & & \{V_2, \neg V_1\} \arrow[dl] & \{\neg V_n, \neg V_4\} \arrow[dd] & \{\neg V_n, V_4, \neg V_5\} \arrow[d] & \{V_4, V_5\} \arrow[dl] \\
& \{V_1, V_n\} \arrow[dr] & & \{V_n, \neg V_1\} \arrow[dl] & & & \{\neg V_n, V_4\} \arrow[dl] & \\
& & \{V_n\} & & & \{\neg V_n\} & &
\end{tikzcd}$$

Res.wid. von $\mathcal{C}$.
Sequenz $(C_1, \dots, C_n)$:
$C_n = C$ und
für alle $1 \leq i \leq n$ gilt
- $C_i \in \mathcal{C}$ oder
- es gibt $j, k < i$ mit $C_i \in Res(C_j, C_k)$.

Resolutionsableitung von $\Box$.

$$\big(\{V_1, V_2\},\ \{\neg V_2, V_n\},\ \{V_1, V_n\},\ \{V_2, \neg V_1\},\ \{V_n, \neg V_1\},\ \{V_n\},$$
$$\{\neg V_n, \neg V_4\},\ \{\neg V_n, V_4, \neg V_5\},\ \{\neg V_n, V_4\}, \{V_4, V_5\},\ \{\neg V_n\},\ \Box\big)$$

Resolvente.
C_1, C_2 Klauseln,
L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.
Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Beispiel einer Resolutionswiderlegung

$$\varphi := (V_1 \vee V_2) \wedge (\neg V_2 \vee V_n) \wedge (V_2 \vee \neg V_1) \wedge (\neg V_n \vee \neg V_4) \wedge (\neg V_n \vee V_4 \vee \neg V_5) \wedge (V_4 \vee V_5)$$

Klauselform.

$$\begin{tikzcd}
\{V_1, V_2\} \arrow[dr] & & \{\neg V_2, V_n\} \arrow[dl] \arrow[dr] & & \{V_2, \neg V_1\} \arrow[dl] & & \{\neg V_n, \neg V_4\} \arrow[ddr] & & \{\neg V_n, V_4, \neg V_5\} \arrow[dr] & & \{V_4, V_5\} \arrow[dl] \\
& \{V_1, V_n\} \arrow[dr] & & \{V_n, \neg V_1\} \arrow[dl] & & & & & & \{\neg V_n, V_4\} \arrow[dll] & \\
& & \{V_n\} \arrow[drrr] & & & & & \{\neg V_n\} \arrow[dll] & & & \\
& & & & & \Box & & & & &
\end{tikzcd}$$

Resolutionsableitung von $\Box$.

$$\Big(\{V_1, V_2\}, \quad \{\neg V_2, V_n\}, \quad \{V_1, V_n\}, \quad \{V_2, \neg V_1\}, \quad \{V_n, \neg V_1\}, \quad \{V_n\},$$
$$\{\neg V_n, \neg V_4\}, \quad \{\neg V_n, V_4, \neg V_5\}, \quad \{\neg V_n, V_4\}, \{V_4, V_5\}, \quad \{\neg V_n\}, \quad \Box \Big)$$

Res.wid. von $\mathcal{C}$.
Sequenz $(C_1, \dots, C_n)$:
$C_n = C$ und
für alle $1 \leq i \leq n$ gilt
- $C_i \in \mathcal{C}$ oder
- es gibt $j, k < i$ mit $C_i \in Res(C_j, C_k)$.

Resolvente.
C_1, C_2 Klauseln,

L Literal
mit $L \in C_1$, $\overline{L} \in C_2$.

Resolvente $C = C_1 \setminus \{L\} \cup C_2 \setminus \{\overline{L}\}$.

Der aussagenlogische Resolutionskalkül

Theorem. Eine Menge $\mathcal{C}$ von Klauseln hat genau dann eine Resolutionswiderlegung, wenn $\mathcal{C}$ unerfüllbar ist. D.h. es gilt

$$\mathcal{C} \vdash_R \square \qquad \text{gdw.} \qquad \mathcal{C} \text{ ist unerfüllbar} \qquad \text{gdw.} \qquad \mathcal{C} \models \bot$$

Der aussagenlogische Resolutionskalkül

Theorem. Eine Menge $\mathcal{C}$ von Klauseln hat genau dann eine Resolutionswiderlegung, wenn $\mathcal{C}$ unerfüllbar ist. D.h. es gilt

$$\mathcal{C} \vdash_R \Box \qquad \text{gdw.} \qquad \mathcal{C} \text{ ist unerfüllbar} \qquad \text{gdw.} \qquad \mathcal{C} \models \bot$$

Lemma. (Korrektheit des Resolutionskalküls)
Wenn eine Menge $\mathcal{C}$ von Klauseln eine Resolutionswiderlegung hat, ist $\mathcal{C}$ unerfüllbar.

Lemma. (Vollständigkeit des Resolutionskalküls)
Jede unerfüllbare Klauselmenge $\mathcal{C}$ hat eine Resolutionswiderlegung.