



# FORMALE SYSTEME

## 22. Vorlesung: Äquivalenzen und Normalformen

Markus Krötzsch

Professur für Wissensbasierte Systeme

TU Dresden, 11. Januar 2024

# Rückblick

# Aussagenlogik

## Syntax:

$$F \rightarrow \mathbf{P} \mid \neg F \mid (F \wedge F) \mid (F \vee F) \mid (F \rightarrow F) \mid (F \leftrightarrow F)$$

## Semantik:

- Wertzuweisungen  $w : \mathbf{P} \rightarrow \{1, 0\}$  zur Interpretation von Atomen
- Erweiterung von Atomen auf Formeln:

$w(F)$	$w(\neg F)$
0	1
1	0

$w(F)$	$w(G)$	$w(F \wedge G)$	$w(F \vee G)$	$w(F \rightarrow G)$	$w(F \leftrightarrow G)$
0	0	0	0	1	1
1	0	0	1	0	0
0	1	0	1	1	0
1	1	1	1	1	1

# Logische Schlussfolgerung

## Modelle:

$w \models F$  gdw.  $w(F) = 1$  gdw. „ $w$  ist Modell von  $F$ “ gdw. „ $w$  erfüllt  $F$ “

## Logische Konsequenzen:

- $\mathcal{F} \models G$  gdw.
- jedes Modell von  $\mathcal{F}$  ist auch ein Modell von  $G$  gdw.
- $G$  ist immer wahr wenn alle Formeln in  $\mathcal{F}$  wahr sind

# Logische Schlussfolgerung

## Modelle:

$w \models F$  gdw.  $w(F) = 1$  gdw. „ $w$  ist Modell von  $F$ “ gdw. „ $w$  erfüllt  $F$ “

## Logische Konsequenzen:

- $\mathcal{F} \models G$  gdw.
- jedes Modell von  $\mathcal{F}$  ist auch ein Modell von  $G$  gdw.
- $G$  ist immer wahr wenn alle Formeln in  $\mathcal{F}$  wahr sind

## Dualität von Modellen und Formeln:

- Je mehr Formeln in  $\mathcal{F}$
- desto weniger Modelle erfüllen alle Formeln in  $\mathcal{F}$
- desto mehr Formeln sind in allen diesen Modellen wahr
- desto mehr Konsequenzen hat  $\mathcal{F}$

$\rightsquigarrow$  Aussagenlogik ist **monoton** (mehr Annahmen  $\Rightarrow$  mehr Schlüsse)

# Modellierungsbeispiel: Sudoku (1)

Sudoku ist ein bekanntes Zahlenpuzzle.

		<b>3</b>						
			<b>1 3</b>		<b>7</b>			
<b>6 1</b>			<b>9</b>					
<b>2</b>		<b>1</b>			<b>8</b>			<b>7</b>
		<b>6</b>		<b>2</b>		<b>4</b>		
<b>5</b>			<b>9</b>			<b>1</b>		<b>3</b>
				<b>4</b>			<b>8 6</b>	
		<b>5</b>		<b>8 7</b>				
						<b>9</b>		

Aufgabe:

- Fülle jedes Feld mit einer Ziffer von 1 bis 9, so dass
- Spalten, Zeilen und die fetten Teilquadrate jede Ziffer nur einmal enthalten

# Modellierungsbeispiel: Sudoku (2)

Wir können Sudoku aussagenlogisch modellieren:

## Modellierungsbeispiel: Sudoku (2)

Wir können Sudoku aussagenlogisch modellieren:

- **Atome:** Für jede Ziffer  $z \in \{1, \dots, 9\}$  und alle Koordinaten  $i, j \in \{1, \dots, 9\}$  verwenden wir ein Atom  $p_z[i, j]$  für die Aussage „an Position  $(i, j)$  steht die Ziffer  $z$ “

## Modellierungsbeispiel: Sudoku (2)

Wir können Sudoku aussagenlogisch modellieren:

- **Atome:** Für jede Ziffer  $z \in \{1, \dots, 9\}$  und alle Koordinaten  $i, j \in \{1, \dots, 9\}$  verwenden wir ein Atom  $p_z[i, j]$  für die Aussage „an Position  $(i, j)$  steht die Ziffer  $z$ “
- **Spielregeln** modellieren wir als logische Formeln:

Für alle  $i, j$ :  $p_1[i, j] \vee p_2[i, j] \vee \dots \vee p_9[i, j]$

Für alle  $i, j, z, z'$  mit  $z \neq z'$ :  $p_z[i, j] \rightarrow \neg p_{z'}[i, j]$

Für alle  $i, i', j, z$  mit  $i \neq i'$ :  $p_z[i, j] \rightarrow \neg p_z[i', j]$

Für alle  $i, j, j', z$  mit  $j \neq j'$ :  $p_z[i, j] \rightarrow \neg p_z[i, j']$

Für alle  $i, j, i', j', z$  mit  $(i, j) \neq (i', j')$ ;

$(i, j), (i', j')$  im gleichen Teilquadrat:  $p_z[i, j] \rightarrow \neg p_z[i', j']$

## Modellierungsbeispiel: Sudoku (2)

Wir können Sudoku aussagenlogisch modellieren:

- **Atome:** Für jede Ziffer  $z \in \{1, \dots, 9\}$  und alle Koordinaten  $i, j \in \{1, \dots, 9\}$  verwenden wir ein Atom  $p_z[i, j]$  für die Aussage „an Position  $(i, j)$  steht die Ziffer  $z$ “
- **Spielregeln** modellieren wir als logische Formeln:

Für alle  $i, j$ :  $p_1[i, j] \vee p_2[i, j] \vee \dots \vee p_9[i, j]$

Für alle  $i, j, z, z'$  mit  $z \neq z'$ :  $p_z[i, j] \rightarrow \neg p_{z'}[i, j]$

Für alle  $i, i', j, z$  mit  $i \neq i'$ :  $p_z[i, j] \rightarrow \neg p_z[i', j]$

Für alle  $i, j, j', z$  mit  $j \neq j'$ :  $p_z[i, j] \rightarrow \neg p_z[i, j']$

Für alle  $i, j, i', j', z$  mit  $(i, j) \neq (i', j')$ ;

$(i, j), (i', j')$  im gleichen Teilquadrat:  $p_z[i, j] \rightarrow \neg p_z[i', j']$

- **Vorgegebene Ziffern** werden als atomare Formeln dargestellt, z.B.  $p_3[3, 1]$  im vorigen Beispiel

## Modellierungsbeispiel: Sudoku (2)

Wir können Sudoku aussagenlogisch modellieren:

- **Atome:** Für jede Ziffer  $z \in \{1, \dots, 9\}$  und alle Koordinaten  $i, j \in \{1, \dots, 9\}$  verwenden wir ein Atom  $p_z[i, j]$  für die Aussage „an Position  $(i, j)$  steht die Ziffer  $z$ “
- **Spielregeln** modellieren wir als logische Formeln:

Für alle  $i, j$ :  $p_1[i, j] \vee p_2[i, j] \vee \dots \vee p_9[i, j]$

Für alle  $i, j, z, z'$  mit  $z \neq z'$ :  $p_z[i, j] \rightarrow \neg p_{z'}[i, j]$

Für alle  $i, i', j, z$  mit  $i \neq i'$ :  $p_z[i, j] \rightarrow \neg p_z[i', j]$

Für alle  $i, j, j', z$  mit  $j \neq j'$ :  $p_z[i, j] \rightarrow \neg p_z[i, j']$

Für alle  $i, j, i', j', z$  mit  $(i, j) \neq (i', j')$ ;

$(i, j), (i', j')$  im gleichen Teilquadrat:  $p_z[i, j] \rightarrow \neg p_z[i', j']$

- **Vorgegebene Ziffern** werden als atomare Formeln dargestellt, z.B.  $p_3[3, 1]$  im vorigen Beispiel

↪ Modelle der Formelmengende entsprechen Lösungen des Sudoku

# Äquivalenzen

# Logische Äquivalenz

Formeln sind äquivalent, wenn sie die gleiche Semantik haben:

Zwei Formeln  $F$  und  $G$  sind **semantisch äquivalent**, in Symbolen  $F \equiv G$ , wenn sie genau die selben Modelle haben, d.h. wenn

für alle Wertzuweisungen  $w$  gilt:  $w(F) = w(G)$

# Logische Äquivalenz

Formeln sind äquivalent, wenn sie die gleiche Semantik haben:

Zwei Formeln  $F$  und  $G$  sind **semantisch äquivalent**, in Symbolen  $F \equiv G$ , wenn sie genau die selben Modelle haben, d.h. wenn

für alle Wertzuweisungen  $w$  gilt:  $w(F) = w(G)$

Beispiel:  $p \rightarrow q \equiv \neg p \vee q$ , wie man mithilfe der Wahrheitwertetabelle zeigen kann (die Spalten der Formeln sind gleich):

$p$	$q$	$p \rightarrow q$	$\neg p$	$\neg p \vee q$	*
0	0	1	1	1	
1	0	0	0	0	
0	1	1	1	1	
1	1	1	0	1	

\* Vereinfachung: Wir beschriften Tabellenspalten ab jetzt nur mit  $F$  statt  $w(F)$ .

# Logische Äquivalenz

Formeln sind äquivalent, wenn sie die gleiche Semantik haben:

Zwei Formeln  $F$  und  $G$  sind **semantisch äquivalent**, in Symbolen  $F \equiv G$ , wenn sie genau die selben Modelle haben, d.h. wenn

für alle Wertzuweisungen  $w$  gilt:  $w(F) = w(G)$

Beispiel:  $p \rightarrow q \equiv \neg p \vee q$ , wie man mithilfe der Wahrheitwertetabelle zeigen kann (die Spalten der Formeln sind gleich):

$p$	$q$	$p \rightarrow q$	$\neg p$	$\neg p \vee q$	*
0	0	1	1	1	
1	0	0	0	0	
0	1	1	1	1	
1	1	1	0	1	

\* Vereinfachung: Wir beschriften Tabellenspalten ab jetzt nur mit  $F$  statt  $w(F)$ .

# Nützliche Eigenschaften von $\equiv$

**Satz:**  $\equiv$  ist eine Äquivalenzrelation, d.h. reflexiv, symmetrisch und transitiv.

**Beweis:**  $\equiv$  ist definiert als die Gleichheit der Modellmengen. Die gesuchten Eigenschaften ergeben sich, da auch die Relation  $=$  auf Mengen eine Äquivalenzrelation ist. □

Weitere Eigenschaften folgen direkt aus den Definitionen:

**Satz:**

- Alle Tautologien sind semantisch äquivalent
- Alle unerfüllbaren Formeln sind semantisch äquivalent

**Satz:** Semantische Äquivalenz entspricht wechselseitiger logischer Konsequenz:

$$F \equiv G \quad \text{genau dann wenn} \quad F \models G \text{ und } G \models F$$

## Nützliche Eigenschaften von $\equiv$

**Satz (Ersetzungstheorem):** Sei  $F$  eine Formel mit einer Teilformel  $G$ . Wenn  $G \equiv G'$  und wenn  $F'$  aus  $F$  gebildet werden kann, indem man ein beliebiges Vorkommen von  $G$  in  $F$  durch  $G'$  ersetzt, dann gilt auch  $F \equiv F'$ .

## Nützliche Eigenschaften von $\equiv$

**Satz (Ersetzungstheorem):** Sei  $F$  eine Formel mit einer Teilformel  $G$ . Wenn  $G \equiv G'$  und wenn  $F'$  aus  $F$  gebildet werden kann, indem man ein beliebiges Vorkommen von  $G$  in  $F$  durch  $G'$  ersetzt, dann gilt auch  $F \equiv F'$ .

**Beweisskizze\*:** Aus  $G \equiv G'$  folgt, dass  $w \models G$  gdw.  $w \models G'$  für beliebige Wertzuweisungen  $w$  gilt. In der rekursiven Definition von  $\models$  erfüllt daher die Teilformel  $G$  genau die selben Bedingungen wie  $G'$ , so dass sich auch für  $F$  und  $F'$  der selbe Wert ergibt. □

Dieser Satz ist ein Sonderfall der allgemeingültigen und intuitiven Tatsache, dass der Wert von rekursiv definierten Funktionen gleich bleibt, wenn man die syntaktische Definition einer Teilberechnung durch eine andere ersetzt, die stets den gleichen Wert liefert.

\* Streng genommen folgt aus der Definition von  $\models$  zunächst nur, dass die Ersetzung einer direkten Unterformel Äquivalenz erhält. Zum Beispiel folgt  $G \wedge H \equiv G' \wedge H$  aus  $G \equiv G'$ . Um zu zeigen, dass auch Ersetzungen in beliebiger Tiefe Äquivalenz erhalten, ist noch ein induktives Argument nötig, welches die Äquivalenz im Formelbaum „nach oben“ propagiert. Aus  $G \wedge H \equiv G' \wedge H$  folgt z.B.  $(G \wedge H) \vee J \equiv (G' \wedge H) \vee J$  usw. usf.

# Junktoren äquivalent ausdrücken

Viele Junktoren können durch andere ausgedrückt werden:

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

$$F \leftrightarrow G \equiv (F \rightarrow G) \wedge (G \rightarrow F) \equiv (F \wedge G) \vee (\neg F \wedge \neg G)$$

$$F \wedge G \equiv \neg(\neg F \vee \neg G) \quad (\text{De Morgansches Gesetz})$$

$$F \vee G \equiv \neg(\neg F \wedge \neg G) \quad (\text{De Morgansches Gesetz})$$

# Junktoren äquivalent ausdrücken

Viele Junktoren können durch andere ausgedrückt werden:

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

$$F \leftrightarrow G \equiv (F \rightarrow G) \wedge (G \rightarrow F) \equiv (F \wedge G) \vee (\neg F \wedge \neg G)$$

$$F \wedge G \equiv \neg(\neg F \vee \neg G) \quad (\text{De Morgansches Gesetz})$$

$$F \vee G \equiv \neg(\neg F \wedge \neg G) \quad (\text{De Morgansches Gesetz})$$

Weitere Junktoren können mithilfe äquivalenter Formeln definiert werden:

$$F \uparrow G \equiv \neg(F \wedge G) \equiv \neg F \vee \neg G \quad (\text{NAND})$$

$$F \downarrow G \equiv \neg(F \vee G) \equiv \neg F \wedge \neg G \quad (\text{NOR})$$

$$\top \equiv \neg p \vee p \equiv p \rightarrow p \quad (\text{Wahrheit; } p \in \mathbf{P} \text{ beliebig})$$

$$\perp \equiv \neg p \wedge p \quad (\text{Falschheit; } p \in \mathbf{P} \text{ beliebig})$$

Wir werden insbesondere  $\top$  und  $\perp$  manchmal verwenden.

## Junktoren äquivalent ausdrücken (2)

**Satz:** Sei  $F$  eine beliebige aussagenlogische Formel.

- Es gibt eine zu  $F$  äquivalente Formel, die nur die Junktoren  $\wedge$  und  $\neg$  enthält.
- Es gibt eine zu  $F$  äquivalente Formel, die nur die Junktoren  $\vee$  und  $\neg$  enthält.

## Junktoren äquivalent ausdrücken (2)

**Satz:** Sei  $F$  eine beliebige aussagenlogische Formel.

- Es gibt eine zu  $F$  äquivalente Formel, die nur die Junktoren  $\wedge$  und  $\neg$  enthält.
- Es gibt eine zu  $F$  äquivalente Formel, die nur die Junktoren  $\vee$  und  $\neg$  enthält.

**Beweis:** Für den ersten Fall ersetzen wir iterativ Teilformeln, die nicht die gewünschte Form haben, gemäß den folgenden Regeln:

$$(G \vee H) \quad \mapsto \quad \neg(\neg G \wedge \neg H)$$

$$(G \rightarrow H) \quad \mapsto \quad \neg(G \wedge \neg H)$$

$$(G \leftrightarrow H) \quad \mapsto \quad (\neg(G \wedge \neg H) \wedge \neg(H \wedge \neg G))$$

Jede Ersetzung führt zu einer äquivalenten Formel (Ersetzungstheorem), der Algorithmus terminiert (jeder Junktor wird höchstens einmal ersetzt) und das Ergebnis hat die gewünschte Form. Der zweite Fall ist analog. □

# Nützliche Äquivalenzen (1)

$$F \wedge G \equiv G \wedge F$$

$$F \vee G \equiv G \vee F$$

Kommutativität

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

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

Assoziativität

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

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

Distributivität

$$F \wedge F \equiv F$$

$$F \vee F \equiv F$$

Idempotenz

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

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

Absorption

# Nützliche Äquivalenzen (2)

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

doppelte Negation

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

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

De Morgansche Gesetze

$$F \wedge \top \equiv F$$

$$F \vee \top \equiv \top$$

Gesetze mit  $\top$

$$F \wedge \perp \equiv \perp$$

$$F \vee \perp \equiv F$$

Gesetze mit  $\perp$

$$\neg\top \equiv \perp$$

$$\neg\perp \equiv \top$$

Alle diese Äquivalenzen können leicht mit Wahrheitstabellen überprüft werden.

# Äquivalente Mengen von Formeln

Die Definition von Äquivalenz ist leicht auf Formelmengen erweiterbar:

Zwei Formelmengen  $\mathcal{F}$  und  $\mathcal{G}$  sind äquivalent, in Symbolen  $\mathcal{F} \equiv \mathcal{G}$ , wenn sie die selben Modelle haben.

# Äquivalente Mengen von Formeln

Die Definition von Äquivalenz ist leicht auf Formelmengen erweiterbar:

Zwei Formelmengen  $\mathcal{F}$  und  $\mathcal{G}$  sind äquivalent, in Symbolen  $\mathcal{F} \equiv \mathcal{G}$ , wenn sie die selben Modelle haben.

Formelmengen verallgemeinern die Konjunktion, denn es gilt:

$$\{F_1, \dots, F_n\} \equiv \{F_1 \wedge \dots \wedge F_n\}$$

**Vereinfachung:** Dank Assoziativität verzichten wir ab jetzt in Formeln wie  $((F_1 \wedge F_2) \wedge F_3) \wedge F_4$  auf Klammern (ebenso für  $\vee$ ).

Allerdings dürfen Formelmengen auch unendlich sein (Konjunktionen dagegen nicht).

$\models$  entspricht  $\rightarrow$  und  $\equiv$  entspricht  $\leftrightarrow$

**Satz (Deduktionstheorem):** Für jede Formelmenge  $\mathcal{F}$  und Formeln  $G$  und  $H$  gilt  $\mathcal{F} \models G \rightarrow H$  genau dann wenn  $\mathcal{F} \cup \{G\} \models H$ .

$\models$  entspricht  $\rightarrow$  und  $\equiv$  entspricht  $\leftrightarrow$

**Satz (Deduktionstheorem):** Für jede Formelmenge  $\mathcal{F}$  und Formeln  $G$  und  $H$  gilt  $\mathcal{F} \models G \rightarrow H$  genau dann wenn  $\mathcal{F} \cup \{G\} \models H$ .

**Beweis:** Einfache Anwendung der Definitionen. ( $\Rightarrow$ ) Wenn jedes Modell von  $\mathcal{F}$  die Formel  $G \rightarrow H$  erfüllt, dann muss jedes dieser Modelle, welches zudem  $G$  erfüllt,  $H$  erfüllen. ( $\Leftarrow$ ) Wenn jedes Modell von  $\mathcal{F}$ , welches zudem  $G$  erfüllt, auch  $H$  erfüllt, dann muss jedes Modell von  $\mathcal{F}$  auch  $G \rightarrow H$  erfüllen. □

$\models$  entspricht  $\rightarrow$  und  $\equiv$  entspricht  $\leftrightarrow$

**Satz (Deduktionstheorem):** Für jede Formelmenge  $\mathcal{F}$  und Formeln  $G$  und  $H$  gilt  $\mathcal{F} \models G \rightarrow H$  genau dann wenn  $\mathcal{F} \cup \{G\} \models H$ .

**Beweis:** Einfache Anwendung der Definitionen. ( $\Rightarrow$ ) Wenn jedes Modell von  $\mathcal{F}$  die Formel  $G \rightarrow H$  erfüllt, dann muss jedes dieser Modelle, welches zudem  $G$  erfüllt,  $H$  erfüllen. ( $\Leftarrow$ ) Wenn jedes Modell von  $\mathcal{F}$ , welches zudem  $G$  erfüllt, auch  $H$  erfüllt, dann muss jedes Modell von  $\mathcal{F}$  auch  $G \rightarrow H$  erfüllen.  $\square$

**Notation:** Wir schreiben  $\models F$  statt  $\emptyset \models F$  um auszudrücken, dass  $F$  allgemeingültig ist.

**Satz:**  $F \equiv G$  genau dann wenn  $\models F \leftrightarrow G$ .

$\models$  entspricht  $\rightarrow$  und  $\equiv$  entspricht  $\leftrightarrow$

**Satz (Deduktionstheorem):** Für jede Formelmenge  $\mathcal{F}$  und Formeln  $G$  und  $H$  gilt  $\mathcal{F} \models G \rightarrow H$  genau dann wenn  $\mathcal{F} \cup \{G\} \models H$ .

**Beweis:** Einfache Anwendung der Definitionen. ( $\Rightarrow$ ) Wenn jedes Modell von  $\mathcal{F}$  die Formel  $G \rightarrow H$  erfüllt, dann muss jedes dieser Modelle, welches zudem  $G$  erfüllt,  $H$  erfüllen. ( $\Leftarrow$ ) Wenn jedes Modell von  $\mathcal{F}$ , welches zudem  $G$  erfüllt, auch  $H$  erfüllt, dann muss jedes Modell von  $\mathcal{F}$  auch  $G \rightarrow H$  erfüllen. □

**Notation:** Wir schreiben  $\models F$  statt  $\emptyset \models F$  um auszudrücken, dass  $F$  allgemeingültig ist.

**Satz:**  $F \equiv G$  genau dann wenn  $\models F \leftrightarrow G$ .

**Beweis:**  $F \equiv G$  gdw.  $F \models G$  und  $G \models F$  gdw.  $\models F \rightarrow G$  und  $\models G \rightarrow F$  (wegen Deduktionstheorem) gdw.  $\models F \leftrightarrow G$ . □

# Curry

Mit Deduktionstheorem und dem Bezug von Formelmengen und Konjunktionen erhält man ein weiteres interessantes Ergebnis:

$$\begin{aligned} \models (F \wedge G) \rightarrow H & \text{ gdw. } \{F \wedge G\} \models H & \text{ gdw. } \{F, G\} \models H \\ & \text{ gdw. } \{F\} \models G \rightarrow H & \text{ gdw. } \models F \rightarrow (G \rightarrow H) \end{aligned}$$

# Curry

Mit Deduktionstheorem und dem Bezug von Formelmengen und Konjunktionen erhält man ein weiteres interessantes Ergebnis:

$$\begin{aligned} \models (F \wedge G) \rightarrow H & \text{ gdw. } \{F \wedge G\} \models H & \text{ gdw. } \{F, G\} \models H \\ & \text{ gdw. } \{F\} \models G \rightarrow H & \text{ gdw. } \models F \rightarrow (G \rightarrow H) \end{aligned}$$

Tatsächlich gilt allgemein:

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

Die Ersetzung von  $(F \wedge G) \rightarrow H$  durch  $F \rightarrow (G \rightarrow H)$  wird **Currying** (im Deutschen manchmal auch: **Schönfinkeln**) genannt

↪ vgl. **funktionale Programmierung**

# Normalformen

# Normalform

**Normalformen** sind besondere syntaktische Formen aussagenlogischer Formeln

Sie heißen so, weil sich jede aussagenlogische Formel in eine äquivalente Formel in Normalform umformen lässt

Wir werden hier drei wichtige Normalformen kennenlernen:

- Negationsnormalform (NNF)
- Konjunktive Normalform (KNF)
- Disjunktive Normalform (DNF)

# Negationsnormalform

Eine Formel  $F$  ist in **Negationsnormalform (NNF)** wenn

- (a) sie nur die Junktoren  $\wedge$ ,  $\vee$  und  $\neg$  enthält und
- (b) der Junktor  $\neg$  nur direkt vor Atomen vorkommt (d.h. nur in Teilformeln der Form  $\neg p$  mit  $p \in \mathbf{P}$ ).

Formeln, die negierte oder nichtnegierte Atome sind, nennt man **Literale**. In NNF darf Negation also nur in Literalen auftauchen.

# Negationsnormalform

Eine Formel  $F$  ist in **Negationsnormalform (NNF)** wenn

- (a) sie nur die Junktoren  $\wedge$ ,  $\vee$  und  $\neg$  enthält und
- (b) der Junktor  $\neg$  nur direkt vor Atomen vorkommt (d.h. nur in Teilformeln der Form  $\neg p$  mit  $p \in \mathbf{P}$ ).

Formeln, die negierte oder nichtnegierte Atome sind, nennt man **Literale**. In NNF darf Negation also nur in Literalen auftauchen.

## Beispiele:

- $(\neg p \wedge q) \vee (p \wedge \neg q)$  ist in NNF
- $(b \wedge b) \vee \neg(b \wedge b)$  ist nicht in NNF
- $q \vee \neg\neg p$  ist nicht in NNF
- $p \leftrightarrow p$  ist nicht in NNF

# Umwandlung in NNF

Es ist möglich, eine Formel rekursiv in NNF umzuformen.

Dazu ersetzen wir zunächst alle Vorkommen von  $\rightarrow$  und  $\leftrightarrow$  durch äquivalente Formeln (wie zuvor).

Sei  $F$  eine Formel, die nur die Junktoren  $\wedge$ ,  $\vee$  und  $\neg$  enthält. Wir definieren eine Formel  $\text{NNF}(F)$  rekursiv wie folgt:

- $\text{NNF}(p) = p$  falls  $p \in \mathbf{P}$
- $\text{NNF}(F \wedge G) = \text{NNF}(F) \wedge \text{NNF}(G)$
- $\text{NNF}(F \vee G) = \text{NNF}(F) \vee \text{NNF}(G)$
- $\text{NNF}(\neg p) = \neg p$  falls  $p \in \mathbf{P}$
- $\text{NNF}(\neg\neg F) = \text{NNF}(F)$
- $\text{NNF}(\neg(F \wedge G)) = \text{NNF}(\neg F) \vee \text{NNF}(\neg G)$
- $\text{NNF}(\neg(F \vee G)) = \text{NNF}(\neg F) \wedge \text{NNF}(\neg G)$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\underline{((p \rightarrow q) \rightarrow p)} \rightarrow p$$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\underline{((p \rightarrow q) \rightarrow p)} \rightarrow p \equiv \underline{((\neg p \vee q) \rightarrow p)} \rightarrow p$$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\begin{aligned} \underline{((p \rightarrow q) \rightarrow p)} \rightarrow p &\equiv \underline{((\neg p \vee q) \rightarrow p)} \rightarrow p \\ &\equiv \underline{(\neg(\neg p \vee q) \vee p)} \rightarrow p \end{aligned}$$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\begin{aligned} \underline{((p \rightarrow q) \rightarrow p)} \rightarrow p &\equiv \underline{((\neg p \vee q) \rightarrow p)} \rightarrow p \\ &\equiv \underline{(\neg(\neg p \vee q) \vee p)} \rightarrow p \\ &\equiv \neg(\neg(\neg p \vee q) \vee p) \vee p \end{aligned}$$

Anschließend wenden wir NNF an:

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\begin{aligned} \underline{((p \rightarrow q) \rightarrow p)} \rightarrow p &\equiv \underline{((\neg p \vee q) \rightarrow p)} \rightarrow p \\ &\equiv \underline{(\neg(\neg p \vee q) \vee p)} \rightarrow p \\ &\equiv \neg(\neg(\neg p \vee q) \vee p) \vee p \end{aligned}$$

Anschließend wenden wir NNF an:

$$\text{NNF}(\neg(\neg(\neg p \vee q) \vee p) \vee p)$$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\begin{aligned} \underline{((p \rightarrow q) \rightarrow p)} \rightarrow p &\equiv \underline{((\neg p \vee q) \rightarrow p)} \rightarrow p \\ &\equiv \underline{(\neg(\neg p \vee q) \vee p)} \rightarrow p \\ &\equiv \neg(\neg(\neg p \vee q) \vee p) \vee p \end{aligned}$$

Anschließend wenden wir NNF an:

$$\text{NNF}(\neg(\neg(\neg p \vee q) \vee p) \vee p) = \text{NNF}(\neg(\neg(\neg p \vee q) \vee p)) \vee \text{NNF}(p)$$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\begin{aligned} \underline{((p \rightarrow q) \rightarrow p) \rightarrow p} &\equiv \underline{((\neg p \vee q) \rightarrow p) \rightarrow p} \\ &\equiv \underline{(\neg(\neg p \vee q) \vee p) \rightarrow p} \\ &\equiv \neg(\neg(\neg p \vee q) \vee p) \vee p \end{aligned}$$

Anschließend wenden wir NNF an:

$$\begin{aligned} \text{NNF}(\neg(\neg(\neg p \vee q) \vee p) \vee p) &= \text{NNF}(\neg(\neg(\neg p \vee q) \vee p)) \vee \text{NNF}(p) \\ &= (\text{NNF}(\neg\neg(\neg p \vee q)) \wedge \text{NNF}(\neg p)) \vee p \end{aligned}$$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\begin{aligned} \underline{((p \rightarrow q) \rightarrow p)} \rightarrow p &\equiv \underline{((\neg p \vee q) \rightarrow p)} \rightarrow p \\ &\equiv \underline{(\neg(\neg p \vee q) \vee p)} \rightarrow p \\ &\equiv \neg(\neg(\neg p \vee q) \vee p) \vee p \end{aligned}$$

Anschließend wenden wir NNF an:

$$\begin{aligned} \text{NNF}(\neg(\neg(\neg p \vee q) \vee p) \vee p) &= \text{NNF}(\neg(\neg(\neg p \vee q) \vee p)) \vee \text{NNF}(p) \\ &= (\text{NNF}(\neg\neg(\neg p \vee q)) \wedge \text{NNF}(\neg p)) \vee p \\ &= (\text{NNF}(\neg p \vee q) \wedge \neg p) \vee p \end{aligned}$$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\begin{aligned} \underline{((p \rightarrow q) \rightarrow p)} \rightarrow p &\equiv \underline{((\neg p \vee q) \rightarrow p)} \rightarrow p \\ &\equiv \underline{(\neg(\neg p \vee q) \vee p)} \rightarrow p \\ &\equiv \neg(\neg(\neg p \vee q) \vee p) \vee p \end{aligned}$$

Anschließend wenden wir NNF an:

$$\begin{aligned} \text{NNF}(\neg(\neg(\neg p \vee q) \vee p) \vee p) &= \text{NNF}(\neg(\neg(\neg p \vee q) \vee p)) \vee \text{NNF}(p) \\ &= (\text{NNF}(\neg\neg(\neg p \vee q)) \wedge \text{NNF}(\neg p)) \vee p \\ &= (\text{NNF}(\neg p \vee q) \wedge \neg p) \vee p \\ &= ((\text{NNF}(\neg p) \vee \text{NNF}(q)) \wedge \neg p) \vee p \end{aligned}$$

# Beispiel

Wir betrachten die Formel  $((p \rightarrow q) \rightarrow p) \rightarrow p$ .

Zunächst eliminieren wir Vorkommen von  $\rightarrow$  in beliebiger Reihenfolge:

$$\begin{aligned} \underline{((p \rightarrow q) \rightarrow p)} \rightarrow p &\equiv \underline{((\neg p \vee q) \rightarrow p)} \rightarrow p \\ &\equiv \underline{(\neg(\neg p \vee q) \vee p)} \rightarrow p \\ &\equiv \neg(\neg(\neg p \vee q) \vee p) \vee p \end{aligned}$$

Anschließend wenden wir NNF an:

$$\begin{aligned} \text{NNF}(\neg(\neg(\neg p \vee q) \vee p) \vee p) &= \text{NNF}(\neg(\neg(\neg p \vee q) \vee p)) \vee \text{NNF}(p) \\ &= (\text{NNF}(\neg\neg(\neg p \vee q)) \wedge \text{NNF}(\neg p)) \vee p \\ &= (\text{NNF}(\neg p \vee q) \wedge \neg p) \vee p \\ &= ((\text{NNF}(\neg p) \vee \text{NNF}(q)) \wedge \neg p) \vee p \\ &= ((\neg p \vee q) \wedge \neg p) \vee p \end{aligned}$$

## NNF-Definition (Wiederholung)

Sei  $F$  eine Formel, die nur die Junktoren  $\wedge$ ,  $\vee$  und  $\neg$  enthält. Wir definieren eine Formel  $\text{NNF}(F)$  rekursiv wie folgt:

- $\text{NNF}(p) = p$  falls  $p \in \mathbf{P}$
- $\text{NNF}(F \wedge G) = \text{NNF}(F) \wedge \text{NNF}(G)$
- $\text{NNF}(F \vee G) = \text{NNF}(F) \vee \text{NNF}(G)$
- $\text{NNF}(\neg p) = \neg p$  falls  $p \in \mathbf{P}$
- $\text{NNF}(\neg\neg F) = \text{NNF}(F)$
- $\text{NNF}(\neg(F \wedge G)) = \text{NNF}(\neg F) \vee \text{NNF}(\neg G)$
- $\text{NNF}(\neg(F \vee G)) = \text{NNF}(\neg F) \wedge \text{NNF}(\neg G)$

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?  
**Ja**, wie man leicht überprüfen kann (der Fall  $\neg$  ist nach der Form der negierten Formel nochmals in vier Unterfälle aufgespalten).

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?  
**Ja**, wie man leicht überprüfen kann (der Fall  $\neg$  ist nach der Form der negierten Formel nochmals in vier Unterfälle aufgespalten).
- **Terminierung:** Ist sichergestellt, dass die rekursive Berechnung terminiert?

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?  
**Ja**, wie man leicht überprüfen kann (der Fall  $\neg$  ist nach der Form der negierten Formel nochmals in vier Unterfälle aufgespalten).
- **Terminierung:** Ist sichergestellt, dass die rekursive Berechnung terminiert?  
**Ja**, denn NNF wird in jedem Rekursionsschritt auf Formeln angewendet, die insgesamt weniger Junktoren haben als zuvor.

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?  
**Ja**, wie man leicht überprüfen kann (der Fall  $\neg$  ist nach der Form der negierten Formel nochmals in vier Unterfälle aufgespalten).
- **Terminierung:** Ist sichergestellt, dass die rekursive Berechnung terminiert?  
**Ja**, denn NNF wird in jedem Rekursionsschritt auf Formeln angewendet, die insgesamt weniger Junktoren haben als zuvor.
- **Korrektheit:** Ist das Ergebnis der rekursiven Umwandlung in NNF?

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?  
**Ja**, wie man leicht überprüfen kann (der Fall  $\neg$  ist nach der Form der negierten Formel nochmals in vier Unterfälle aufgespalten).
- **Terminierung:** Ist sichergestellt, dass die rekursive Berechnung terminiert?  
**Ja**, denn NNF wird in jedem Rekursionsschritt auf Formeln angewendet, die insgesamt weniger Junktoren haben als zuvor.
- **Korrektheit:** Ist das Ergebnis der rekursiven Umwandlung in NNF?  
**Ja**, denn  $\neg$  kommt im Ergebnis nur in Fall  $\neg p$  ( $p \in \mathbf{P}$ ) vor.

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?  
**Ja**, wie man leicht überprüfen kann (der Fall  $\neg$  ist nach der Form der negierten Formel nochmals in vier Unterfälle aufgespalten).
- **Terminierung:** Ist sichergestellt, dass die rekursive Berechnung terminiert?  
**Ja**, denn NNF wird in jedem Rekursionsschritt auf Formeln angewendet, die insgesamt weniger Junktoren haben als zuvor.
- **Korrektheit:** Ist das Ergebnis der rekursiven Umwandlung in NNF?  
**Ja**, denn  $\neg$  kommt im Ergebnis nur in Fall  $\neg p$  ( $p \in \mathbf{P}$ ) vor.  
Ist das Ergebnis semantisch äquivalent zur Eingabe?

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?  
**Ja**, wie man leicht überprüfen kann (der Fall  $\neg$  ist nach der Form der negierten Formel nochmals in vier Unterfälle aufgespalten).
- **Terminierung:** Ist sichergestellt, dass die rekursive Berechnung terminiert?  
**Ja**, denn NNF wird in jedem Rekursionsschritt auf Formeln angewendet, die insgesamt weniger Junktoren haben als zuvor.
- **Korrektheit:** Ist das Ergebnis der rekursiven Umwandlung in NNF?  
**Ja**, denn  $\neg$  kommt im Ergebnis nur in Fall  $\neg p$  ( $p \in \mathbf{P}$ ) vor.  
Ist das Ergebnis semantisch äquivalent zur Eingabe?  
**Ja**, wie man durch ein induktives Argument leicht zeigen kann  
Induktionsanfang: Äquivalenz gilt für  $p$  und  $\neg p$ ; Induktionshypothese:  $F \equiv \text{NNF}(F)$  und  $G \equiv \text{NNF}(G)$ ; Induktionsschritt: für jeden rekursiven Fall folgt Äquivalenz aus Hypothese, Ersetzungstheorem und bekannten Äquivalenzgesetzen.

# NNF-Umwandlung: Korrektheit

Ist diese Umformung korrekt?

- **Wohldefiniertheit:** Deckt die Rekursion wirklich jeden Fall ab?

**Ja**, wie man leicht überprüfen kann (der Fall  $\neg$  ist nach der Form der negierten Formel nochmals in vier Unterfälle aufgespalten).

- **Terminierung:** Ist sichergestellt, dass die rekursive Berechnung terminiert?

**Ja**, denn NNF wird in jedem Rekursionsschritt auf Formeln angewendet, die insgesamt weniger Junktoren haben als zuvor.

- **Korrektheit:** Ist das Ergebnis der rekursiven Umwandlung in NNF?

**Ja**, denn  $\neg$  kommt im Ergebnis nur in Fall  $\neg p$  ( $p \in \mathbf{P}$ ) vor.

Ist das Ergebnis semantisch äquivalent zur Eingabe?

**Ja**, wie man durch ein induktives Argument leicht zeigen kann

Induktionsanfang: Äquivalenz gilt für  $p$  und  $\neg p$ ; Induktionshypothese:  $F \equiv \text{NNF}(F)$  und  $G \equiv \text{NNF}(G)$ ; Induktionsschritt: für jeden rekursiven Fall folgt Äquivalenz aus Hypothese, Ersetzungstheorem und bekannten Äquivalenzgesetzen.

**Satz:** Jede Formel kann (in linearer Zeit) in eine äquivalente Formel in NNF umgewandelt werden.

# Konjunktive und Disjunktive Normalform

Wir erinnern uns: **Literale** = negierte oder nichtnegierte Atome

Eine Formel  $F$  ist in **konjunktiver Normalform (KNF)** wenn sie eine Konjunktion von Disjunktionen von Literalen ist, d.h. wenn sie die Form hat:

$$(L_{1,1} \vee \dots \vee L_{1,m_1}) \wedge (L_{2,1} \vee \dots \vee L_{2,m_2}) \wedge \dots \wedge (L_{n,1} \vee \dots \vee L_{n,m_n})$$

wobei die Formeln  $L_{i,j}$  Literale sind. Eine Disjunktion von Literalen heißt **Klausel**.

Eine Formel  $F$  ist in **disjunktiver Normalform (DNF)** wenn sie eine Disjunktion von Konjunktionen von Literalen ist, d.h. wenn sie die Form hat:

$$(L_{1,1} \wedge \dots \wedge L_{1,m_1}) \vee (L_{2,1} \wedge \dots \wedge L_{2,m_2}) \vee \dots \vee (L_{n,1} \wedge \dots \wedge L_{n,m_n})$$

wobei die Formeln  $L_{i,j}$  Literale sind. Eine Konjunktion von Literalen heißt **Monom**.

# KNF und DNF bilden (Methode 1)

Man kann KNF und DNF direkt aus der Wahrheitwertetabelle ablesen:

$p$	$q$	$\neg p \leftrightarrow q$
0	0	0
1	0	1
0	1	1
1	1	0

Disjunktive Normalform:

- Für jede Zeile mit Wert **1**, bilde ein Monom mit allen Atomen, wobei genau die Atome mit Wert **0** negiert werden
- Beispiel:  $(p \wedge \neg q) \vee (\neg p \wedge q)$

Konjunktive Normalform:

- Für jede Zeile mit Wert **0**, bilde eine Klausel mit allen Atomen, wobei genau die Atome mit Wert **1** negiert werden
- Beispiel:  $(p \vee q) \wedge (\neg p \vee \neg q)$

## KNF und DNF bilden (Methode 2)

Man kann KNF und DNF bilden, indem man die NNF erzeugt und anschließend Distributivgesetze anwendet  $\leadsto$  oft direkter

### **Konjunktive Normalform**

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

## KNF und DNF bilden (Methode 2)

Man kann KNF und DNF bilden, indem man die NNF erzeugt und anschließend Distributivgesetze anwendet  $\leadsto$  oft direkter

### Konjunktive Normalform

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

#### Beispiel:

$$(p \wedge \neg q) \vee (\neg p \wedge q)$$

## KNF und DNF bilden (Methode 2)

Man kann KNF und DNF bilden, indem man die NNF erzeugt und anschließend Distributivgesetze anwendet  $\leadsto$  oft direkter

### Konjunktive Normalform

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

#### Beispiel:

$$(p \wedge \neg q) \vee (\neg p \wedge q) \equiv ((p \wedge \neg q) \vee \neg p) \wedge ((p \wedge \neg q) \vee q)$$

## KNF und DNF bilden (Methode 2)

Man kann KNF und DNF bilden, indem man die NNF erzeugt und anschließend Distributivgesetze anwendet  $\leadsto$  oft direkter

### Konjunktive Normalform

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

#### Beispiel:

$$\begin{aligned}(p \wedge \neg q) \vee (\neg p \wedge q) &\equiv ((p \wedge \neg q) \vee \neg p) \wedge ((p \wedge \neg q) \vee q) \\ &\equiv (p \vee \neg p) \wedge (\neg q \vee \neg p) \wedge ((p \wedge \neg q) \vee q)\end{aligned}$$

## KNF und DNF bilden (Methode 2)

Man kann KNF und DNF bilden, indem man die NNF erzeugt und anschließend Distributivgesetze anwendet  $\leadsto$  oft direkter

### Konjunktive Normalform

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

#### Beispiel:

$$\begin{aligned}(p \wedge \neg q) \vee (\neg p \wedge q) &\equiv ((p \wedge \neg q) \vee \neg p) \wedge ((p \wedge \neg q) \vee q) \\ &\equiv (p \vee \neg p) \wedge (\neg q \vee \neg p) \wedge ((p \wedge \neg q) \vee q) \\ &\equiv (p \vee \neg p) \wedge (\neg q \vee \neg p) \wedge (p \vee q) \wedge (\neg q \vee q)\end{aligned}$$

## KNF und DNF bilden (Methode 2)

Man kann KNF und DNF bilden, indem man die NNF erzeugt und anschließend Distributivgesetze anwendet  $\leadsto$  oft direkter

### Konjunktive Normalform

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

#### Beispiel:

$$\begin{aligned}(p \wedge \neg q) \vee (\neg p \wedge q) &\equiv ((p \wedge \neg q) \vee \neg p) \wedge ((p \wedge \neg q) \vee q) \\ &\equiv (p \vee \neg p) \wedge (\neg q \vee \neg p) \wedge ((p \wedge \neg q) \vee q) \\ &\equiv (p \vee \neg p) \wedge (\neg q \vee \neg p) \wedge (p \vee q) \wedge (\neg q \vee q)\end{aligned}$$

(Man könnte die wahren Klauseln  $(p \vee \neg p)$  und  $(\neg q \vee q)$  streichen.)

### Disjunktive Normalform

Distributivgesetz:  $F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$  (analog)

# Wie effizient sind die Umformungen in KNF?

## **Ableitung aus Wahrheitwertetabelle**

Anzahl der Zeilen (=Klauseln) **exponentiell** in Variablenzahl

# Wie effizient sind die Umformungen in KNF?

## Ableitung aus Wahrheitwertetabelle

Anzahl der Zeilen (=Klauseln) **exponentiell** in Variablenzahl

## Umformung mit Distributivgesetz

- Negationsnormalform hat lineare Größe
- Distributivgesetz kann Vorkommen von Formeln verdoppeln  $\leadsto$  exponentielles Wachstum

# Wie effizient sind die Umformungen in KNF?

## Ableitung aus Wahrheitwertetabelle

Anzahl der Zeilen (=Klauseln) **exponentiell** in Variablenzahl

## Umformung mit Distributivgesetz

- Negationsnormalform hat lineare Größe
- Distributivgesetz kann Vorkommen von Formeln verdoppeln  $\leadsto$  exponentielles Wachstum

**Beispiel:** Wir betrachten die Atome  $a_i$  und  $b_i$  für  $i \in \{1, \dots, n\}$ . Für

$$(a_1 \wedge b_1) \vee (a_2 \wedge b_2) \vee \dots \vee (a_n \wedge b_n)$$

ergibt sich die KNF:

$$\underbrace{(a_1 \vee a_2 \vee \dots \vee a_n) \wedge (b_1 \vee a_2 \vee \dots \vee a_n) \wedge \dots \wedge (b_1 \vee b_2 \vee \dots \vee b_n)}_{2^n \text{ Klauseln mit allen Kombinationen aus } a_i \text{ und } b_i}$$

$\leadsto$  Anzahl der Klauseln **exponentiell** in Variablenzahl

# Effizientere Normalformen?

Unsere Umformungen sind schlimmstenfalls exponentiell

- Allgemein unvermeidbar: Es gibt oft keine kleinere KNF/DNF
- Für KNF (aber nicht für DNF!) kann das Problem durch Einführung zusätzlicher Hilfsvariablen gelöst werden

(Es gibt eine polynomielle Formel in KNF mit zusätzlichen Variablen, welche „bezüglich der Variablen der ursprünglichen Formel“ semantisch äquivalent ist)

**Beispiel:**  $(a_1 \wedge b_1) \vee (a_2 \wedge b_2) \vee \dots \vee (a_n \wedge b_n)$  ist ausdrückbar als:

$$\begin{aligned} & ((a_1 \wedge b_1) \vee c_1) \\ & \wedge (c_1 \leftrightarrow ((a_2 \wedge b_2) \vee c_2)) \\ & \wedge (c_2 \leftrightarrow ((a_3 \wedge b_3) \vee c_3)) \\ & \dots \wedge (c_{n-1} \leftrightarrow (a_n \wedge b_n)) \end{aligned}$$

Diese Formel ist „fast“ äquivalent (bis auf die zusätzlichen  $c_i$ ) und hat eine polynomielle KNF

(Jede Zeile kann einzeln in KNF übersetzt werden, wobei Formeln fester Größe entstehen, von denen es linear viele gibt.)

# Zusammenfassung und Ausblick

In der Aussagenlogik gelten viele **nützliche Äquivalenzen** die man für Umformungen ausnutzen kann

Die **Negationsnormalform** vereinfacht Formeln, indem Negationen „nach unten“ verschoben werden

**Konjunktive** und **disjunktive Normalform** stellen beliebige Formeln als Konjunktion bzw. Disjunktion dar, allerdings zum Teil mit erheblichen (exponentiellen) Kosten

Offene Fragen:

- Geht logisches Schließen auch ohne Wahrheitwertetabellen?
- Wie (in)effizient ist logisches Schließen?
- Was hat das mit Sprachen, Berechnung und TMs zu tun?