Kapitel L: II

II. Aussagenlogik

- Syntax der Aussagenlogik
- Semantik der Aussagenlogik
- Eigenschaften des Folgerungsbegriffs
- Äquivalenz
- Formeltransformation
- Normalformen

- Bedeutung der Folgerung
- Erfüllbarkeitsalgorithmen
- Semantische Bäume
- Weiterentwicklung semantischer Bäume
- Syntaktische Schlussfolgerungsverfahren
- Erfüllbarkeitsprobleme

Syntax der Aussagenlogik

Definition 1 (Sprache der Aussagenlogik)

$\Sigma = \{A_0, A_1, A_2, \ldots\}$

abzählbar unendliche Menge von Atomen.

Hilfszeichen: $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$ Junktoren

$(\ ,\ )$ Klammern

Definition 2 (Aussagenlogische Formeln über Σ)

1. Primformel.

   Jedes Atom aus Σ ist eine Formel.

2. logisches Nicht.

   Ist α eine Formel, so auch $(\neg\alpha)$.

3. logisches Und, logisches Oder.

   Sind α und β Formeln, so sind es auch $(\alpha \wedge \beta)$ und $(\alpha \vee \beta)$.

4. logische Implikation, logische Äquivalenz.

   Sind α und β Formeln, so sind es auch $(\alpha \rightarrow \beta)$ und $(\alpha \leftrightarrow \beta)$.

5. Nur die mittels (1) - (4) gebildeten Ausdrücke sind Formeln.

Bemerkungen:

- Im allgemeinen Fall beschreibt Σ eine Signatur mit einer Menge von Sorten und einer Menge von Operationen auf diesen Sorten.
- Es handelt sich bei der Menge der Formeln um eine induktiv definierte Menge (Sprachgebrauch auch: rekursive Definition). Neue Elemente werden aus bereits bekannten nach einem festgelegtenen Konstruktionsschema aufgebaut.
- Klammern gehören zur Formel; sie machen ihre Struktur eindeutig.

Beispiele.

Atomvorrat $\Sigma = \{A, B, C, \ldots, P, Q, R, \ldots, A_0, A_1, A_2, \ldots\}$

Formeln:	keine Formeln:
A	(A)
$(\neg A)$	$(A \vee B))$
$((\neg A) \vee B)$	$(A\neg \vee B)$
$(R \wedge (\neg Q))$	$(\wedge Q)$
$(P \wedge (Q \wedge R))$	(PQ)

Syntax der Aussagenlogik

Vereinbarungen:

1. Σ wird nicht mehr explizit festgelegt.
2. $A, B, C, \ldots, P, Q, R, \ldots, A_0, A_1, A_2, \ldots$ bezeichnen Atome.
3. $\alpha, \beta, \gamma, \ldots, \varphi, \psi, \pi, \ldots, \alpha_0, \alpha_1, \alpha_2, \ldots$ bezeichnen Formeln.
4. $\alpha, \beta, \gamma, \ldots, \varphi, \psi, \pi, \ldots, \alpha_0, \alpha_1, \alpha_2, \ldots$ werden auch als Platzhalter zur Beschreibung von Formeln bestimmter Struktur verwendet.

Beispiele.

$(\varphi \rightarrow \varphi)$
$(\varphi \rightarrow A)$
$(B \rightarrow \varphi)$
$((\varphi \rightarrow \psi) \leftrightarrow ((\neg\varphi) \vee \psi))$
$((\varphi \vee \varphi) \rightarrow \varphi)$
$(\varphi \rightarrow (\varphi \vee \varphi))$

Syntax der Aussagenlogik

Definition 3 (*atoms*(α))

Die Funktion *atoms*(α) bezeichnet die Menge der in α vorkommenden Atome und ist wie folgt definiert.

1. *atoms*$(A) := \{A\}$
2. *atoms*$(\neg\alpha) :=$ *atoms*(α)
3. *atoms*$(\alpha \vee \beta) =$ *atoms*$(\alpha \wedge \beta) :=$ *atoms*$(\alpha) \cup$ *atoms*(β)
4. *atoms*$(\alpha \rightarrow \beta) =$ *atoms*$(\alpha \leftrightarrow \beta) :=$ *atoms*$(\alpha) \cup$ *atoms*(β)

Bemerkungen:

- *atoms*(α) ist endlich.
- *atoms*$(\alpha) \subseteq \Sigma$

Syntax der Aussagenlogik

- Bindungsstärke:
  1. $\neg$ bindet stärker als $\wedge$.
  2. $\wedge$ bindet stärker als $\vee$.
  3. $\vee$ bindet stärker als $\rightarrow$ und $\leftrightarrow$
- Klammereinsparung:
  1. $\wedge$, $\vee$, $\leftrightarrow$ werden als linksgeklammert angesehen.
  2. $\neg$ wird als rechtsgeklammert angesehen.
  3. $\rightarrow$ muss immer geklammert werden.
- Mengenschreibweise:
  $\{\alpha, \beta, \gamma, \ldots\}$ soll verstanden werden als $\alpha \wedge \beta \wedge \gamma \wedge \ldots$

Bemerkungen:

- $\wedge$, $\vee$, $\leftrightarrow$ sind assoziativ; $\rightarrow$ ist nicht assoziativ (wird später gezeigt).
- Im Zweifelsfall immer Klammern verwenden, z. B. zur Gruppierung von Aussagen in Äquivalenzen.

Beispiele:

- $(\neg((A \vee (\neg B)) \wedge ((\neg A) \vee ((\neg B) \wedge (\neg(\neg C))))))$ lässt sich schreiben als $\neg((A \vee \neg B) \wedge (\neg A \vee (\neg B \wedge \neg\neg C)))$
- $((((\neg A) \vee B) \vee D) \vee (\neg E))$ lässt sich schreiben als $\neg A \vee B \vee D \vee \neg E$

Syntax der Aussagenlogik

Definition 4 (Formellänge $|\alpha|$)

Die Funktion $|\alpha|$ zählt die Anzahl der Symbole in α und ist wie folgt definiert.

1. $|A|_1 := 1$
2. $|(\neg\alpha)|_1 := |\alpha|_1 + 3$
3. $|(\alpha \wedge \beta)|_1 = |(\alpha \vee \beta)|_1 := |\alpha|_1 + |\beta|_1 + 3$
4. $|(\alpha \rightarrow \beta)|_1 = |(\alpha \leftrightarrow \beta)|_1 := |\alpha|_1 + |\beta|_1 + 3$

Vergröberung:

- $|\alpha|_2$ zählt nur die tatsächlich gesetzten Klammern.
- $|\alpha|_3$ zählt keine Klammern.

Beispiele:

$|\neg(\neg A \vee \neg B)|_1 =$

$|\neg(\neg A \vee \neg B)|_2 =$

$|\neg(\neg A \vee \neg B)|_3 =$

Syntax der Aussagenlogik

Definition 4 (Formellänge $|\alpha|$)

Die Funktion $|\alpha|$ zählt die Anzahl der Symbole in α und ist wie folgt definiert.

1. $|A|_1 := 1$
2. $|(\neg\alpha)|_1 := |\alpha|_1 + 3$
3. $|(\alpha \wedge \beta)|_1 = |(\alpha \vee \beta)|_1 := |\alpha|_1 + |\beta|_1 + 3$
4. $|(\alpha \rightarrow \beta)|_1 = |(\alpha \leftrightarrow \beta)|_1 := |\alpha|_1 + |\beta|_1 + 3$

Vergröberung:

- $|\alpha|_2$ zählt nur die tatsächlich gesetzten Klammern.
- $|\alpha|_3$ zählt keine Klammern.

Beispiele:

$|\neg(\neg A \vee \neg B)|_1 = 14$

$|\neg(\neg A \vee \neg B)|_2 = 8$

$|\neg(\neg A \vee \neg B)|_3 = 6$

Semantik der Aussagenlogik

Definition 5 (Bewertung, Interpretation)

Sei Σ ein Vorrat von Atomen. Eine Abbildung

$$\mathcal{I} : \Sigma \longrightarrow \{0, 1\}$$

heißt Bewertung oder Interpretation der Atome in Σ.

Bemerkungen:

- Atome werden *unabhängig voneinander* bewertet. Die Bewertung eines Atoms hängt nicht von der Bewertung anderer Atome ab.
- Die Atome können *nur* die Wahrheitswerte *wahr „1“* oder *falsch „0“* annehmen. (Zweiwertigkeits- oder Bivalenzprinzip, Satz vom ausgeschlossenen Dritten).
- Atome können *nur einen* der Wahrheitswerte erhalten (Satz vom ausgeschlossenen Widerspruch).
- Angenommen, das Atom A stehe für die Aussage *„Es regnet“*.
  Die Semantik von A besteht in der Zuordnung des Wertes „0“ oder „1“ zu A.
  Wir verknüpfen mit dem Wert „0“ den Sachverhalt, dass die Aussage, für die A steht, falsch ist. Mit dem Wert „1“ verknüpfen wir den Sachverhalt, dass die Aussage, für die A steht, wahr ist.

- Die „wahre“ Semantik von A ist die Semantik der Realität. Sie muss nicht mit der durch $\mathcal{I}$ festgelegten Semantik übereinstimmen.

  $\mathcal{I}$ definiert die Semantik einer *Modellwelt.*

  „$\mathcal{I}(A) = 1$“ heißt, es regnet in unserer Modellwelt bzw. in unserem Modell.

- Die Semantik der Realität ist die in einer Gemeinschaft akzeptierte Zuordnung von Symbolen zu wahrgenommenen Objekten (Pragmatik). Z. B. ist die Semantik von *„Es regnet“*, dass gerade Wasser aus Wolken vom Himmel fällt.

Semantik der Aussagenlogik

Definition 6 (Bewertung aussagenlogischer Formeln)

Sei $\mathcal{I}$ eine Bewertung der Atome in Σ. Die Erweiterung von $\mathcal{I}$ mit

$$\mathcal{I} : \{\alpha \mid \alpha \text{ aussagenlogische Formel über } \Sigma\} \longrightarrow \{0, 1\}$$

gemäß folgender Vereinbarungen heißt Bewertung der aussagenlogischen Formeln über Σ.

- $\mathcal{I}(\neg\alpha) = \begin{cases} 1, & \text{falls } \mathcal{I}(\alpha) = 0 \\ 0, & \text{sonst} \end{cases}$

	$\neg$
α 0	1
1	0

- $\mathcal{I}(\alpha \wedge \beta) = \begin{cases} 1, & \text{falls } \mathcal{I}(\alpha) = \mathcal{I}(\beta) = 1 \\ 0, & \text{sonst} \end{cases}$

	$\wedge$	β 0	1
α	0	0	0
	1	0	1

- $\mathcal{I}(\alpha \vee \beta) = \begin{cases} 1, & \text{falls } \mathcal{I}(\alpha) = 1 \text{ oder } \mathcal{I}(\beta) = 1 \\ 0, & \text{sonst} \end{cases}$

	$\vee$	β 0	1
α	0	0	1
	1	1	1

Semantik der Aussagenlogik

Definition 6 (Bewertung aussagenlogischer Formeln (Fortsetzung))

- $\mathcal{I}(\alpha \rightarrow \beta) = \begin{cases} 1, & \text{falls } \mathcal{I}(\alpha) = 0 \text{ oder } \mathcal{I}(\beta) = 1 \\ 0, & \text{sonst} \end{cases}$

$\rightarrow$	β	0	1
α	0	1	1
	1	0	1

- $\mathcal{I}(\alpha \leftrightarrow \beta) = \begin{cases} 1, & \text{falls } \mathcal{I}(\alpha) = \mathcal{I}(\beta) \\ 0, & \text{sonst} \end{cases}$

$\leftrightarrow$	β	0	1
α	0	1	0
	1	0	1

Bemerkungen:

- Für die Bewertung von Formeln werden nur die Bewertungen der Atome benötigt, die auch in der Formel vorkommen. Stichwort: „Koinzidenztheorem“

  Beispiele:

  - $\alpha = (A \vee B) \wedge (\neg A \vee B) \wedge (A \vee \neg B)$
    $\mathcal{I}(A) = 0, \quad \mathcal{I}(B) = 1, \quad \mathcal{I}$ sonst beliebig.
    $\leadsto \mathcal{I}(\alpha) =$

  - $\beta = A \wedge (A \rightarrow B) \wedge (B \rightarrow C) \wedge C$
    $\mathcal{I}(A) = \mathcal{I}(B) = \mathcal{I}(C) = 1$
    $\leadsto \mathcal{I}(\beta) =$

  - $\gamma = A \vee \neg A$
    $\mathcal{I}(A) = 1$
    $\leadsto \mathcal{I}(\gamma) =$

  - $\delta = A \wedge \neg A$
    $\mathcal{I}(A) = 1$
    $\leadsto \mathcal{I}(\delta) =$

- Die Junktoren der Aussagenlogik haben die folgende wichtige Eigenschaft: Die Bewertung der Atome, die in der Formel vorkommen, legen den Wahrheitswert der Formel eindeutig fest. Solche Junktoren bezeichnet man als *extensionale* Junktoren; *intensionale* Junktoren legen den Wahrheitswert nicht fest. In dem Satz „A hat eine Freundin gefunden, weil er ein teures Parfum benutzt.“ ist der Wahrheitswert der Gesamtaussage nicht abhängig von den Wahrheitswerten der Teilaussagen.

Semantik der Aussagenlogik

Die Formalisierung von Aussagen über die Realität in der Aussagenlogik kann aufgrund der Definiton der Interpretation unmittelbar mit den Junktoren erfolgen.

Beispiel.

Aussage	Formalisierung
„Es regnet.“	A
„Die Straße ist nass.“	B

„Es regnet nicht.“

„Es regnet oder die Straße ist nass.“

„Es regnet und die Straße ist nass.“

„Wenn es regnet, ist die Straße nass.“

„Nur wenn es regnet, ist die Straße nass.“

„Die Straße ist nur dann nass, wenn es regnet.“

„Genau dann, wenn es regnet, ist die Straße nass.“

„Die Straße ist nass genau dann, wenn es regnet.“

„Die Straße ist nass dann und nur dann, wenn es regnet.“

Semantik der Aussagenlogik

Die Formalisierung von Aussagen über die Realität in der Aussagenlogik kann aufgrund der Definiton der Interpretation unmittelbar mit den Junktoren erfolgen.

Beispiel.

Aussage	Formalisierung
„Es regnet.“	A
„Die Straße ist nass.“	B
„Es regnet nicht.“	$\neg A$
„Es regnet oder die Straße ist nass.“	$A \vee B$
„Es regnet und die Straße ist nass.“	$A \wedge B$
„Wenn es regnet, ist die Straße nass.“	$A \rightarrow B$
„Nur wenn es regnet, ist die Straße nass.“	$B \rightarrow A$
„Die Straße ist nur dann nass, wenn es regnet.“	$B \rightarrow A$
„Genau dann, wenn es regnet, ist die Straße nass.“	$A \leftrightarrow B$
„Die Straße ist nass genau dann, wenn es regnet.“	$A \leftrightarrow B$
„Die Straße ist nass dann und nur dann, wenn es regnet.“	$A \leftrightarrow B$

Semantik der Aussagenlogik

Satz 1 (Koinzidenztheorem)

Seien $\mathcal{I}_1$ und $\mathcal{I}_2$ zwei Interpretationen für Σ und es gelte für alle Atome A in *atoms*(α) einer aussagenlogischen Formel α:

$$\mathcal{I}_1(A) = \mathcal{I}_2(A), \quad \text{dann gilt auch:} \quad \mathcal{I}_1(\alpha) = \mathcal{I}_2(\alpha)$$

Beweisidee

Durch Induktion über den Formelaufbau einer Formel α.

Bemerkungen:

- Rechtfertigung des Koinzidenztheorems:
  Beachte dass α bzgl. *atoms*$(\alpha) \subseteq \Sigma$ definiert ist, $\mathcal{I}$ bzgl. Σ.

- Bestimmung aller Wahrheitswerte von α:
  1. Betrachte die Menge aller möglichen Interpretationen.
  2. Fasse jeweils alle Interpretation $\mathcal{I}$ zu einer Klasse zusammen, die bzgl. der Atome in α identisch sind. Dies entspricht einer Einschränkung der $\mathcal{I}$ auf *atoms*(α) bzw. der Projektion auf *atoms*(α).
     Begründung: Die Bewertung von α hängt nur von der Bewertung der in α vorkommenden Atome ab.
  3. Wähle (beliebig) aus jeder Klasse eine Interpretation. Offensichtlich ist der Wahrheitswert von α bestimmbar und es gilt:
     - Es gibt pro Atom 2 Wahrheitswerte.
     - $|$*atoms*$(\alpha)| = n \leadsto 2^n$ Klassen von Interpretationen
     - Systematische Aufzählung in Tabellenform $\leadsto$ Wahrheitstafel

Semantik der Aussagenlogik

Definition 7 (Wahrheitstafel)

Sei α eine aussagenlogische Formel mit $|\mathit{atoms}(\alpha)| = n$.

Die Tabelle mit den 2^n möglichen Bewertungen der Atome von α und den zugehörigen Bewertungen von α heißt Wahrheitstafel für α.

Beispiel:

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

A	B	C	$\neg C$	$A \vee B$	α
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Semantik der Aussagenlogik

Definition 7 (Wahrheitstafel)

Sei α eine aussagenlogische Formel mit $|\mathit{atoms}(\alpha)| = n$.

Die Tabelle mit den 2^n möglichen Bewertungen der Atome von α und den zugehörigen Bewertungen von α heißt Wahrheitstafel für α.

Beispiel:

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

A	B	C	$\neg C$	$A \vee B$	α
0	0	0	1	0	0
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Semantik der Aussagenlogik

Definition 7 (Wahrheitstafel)

Sei α eine aussagenlogische Formel mit $|\mathit{atoms}(\alpha)| = n$.

Die Tabelle mit den 2^n möglichen Bewertungen der Atome von α und den zugehörigen Bewertungen von α heißt Wahrheitstafel für α.

Beispiel:

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

A	B	C	$\neg C$	$A \vee B$	α
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Semantik der Aussagenlogik

Definition 7 (Wahrheitstafel)

Sei α eine aussagenlogische Formel mit $|\mathit{atoms}(\alpha)| = n$.

Die Tabelle mit den 2^n möglichen Bewertungen der Atome von α und den zugehörigen Bewertungen von α heißt Wahrheitstafel für α.

Beispiel:

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

A	B	C	$\neg C$	$A \vee B$	α
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	1	1
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Semantik der Aussagenlogik

Definition 7 (Wahrheitstafel)

Sei α eine aussagenlogische Formel mit $|\mathit{atoms}(\alpha)| = n$.

Die Tabelle mit den 2^n möglichen Bewertungen der Atome von α und den zugehörigen Bewertungen von α heißt Wahrheitstafel für α.

Beispiel:

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

A	B	C	$\neg C$	$A \vee B$	α
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	1	1
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	0	1	0
1	1	0	1	1	1
1	1	1	0	1	0

Semantik der Aussagenlogik

Definition 8 (Erfüllbarkeitsbegriffe)

Sei α eine aussagenlogische Formel. Dann seien folgende Erfüllbarkeitsbegriffe vereinbart.

1. α erfüllbar $:\Leftrightarrow$ Es gibt $\mathcal{I}$ mit $\mathcal{I}(\alpha) = 1$.
2. α falsifizierbar $:\Leftrightarrow$ Es gibt $\mathcal{I}$ mit $\mathcal{I}(\alpha) = 0$.
3. α tautologisch $:\Leftrightarrow$ Für alle $\mathcal{I}$ gilt $\mathcal{I}(\alpha) = 1$.
4. α widerspruchsvoll $:\Leftrightarrow$ Für alle $\mathcal{I}$ gilt $\mathcal{I}(\alpha) = 0$.

Beispiele:

	erfüllbar	falsifizierbar	tautologisch	widerspruchsvoll
$\alpha = A \wedge \neg A$				
$\alpha = A \vee \neg A$				
$\alpha = A$				

Semantik der Aussagenlogik

Definition 8 (Erfüllbarkeitsbegriffe)

Sei α eine aussagenlogische Formel. Dann seien folgende Erfüllbarkeitsbegriffe vereinbart.

1. α erfüllbar $:\Leftrightarrow$ Es gibt $\mathcal{I}$ mit $\mathcal{I}(\alpha) = 1$.
2. α falsifizierbar $:\Leftrightarrow$ Es gibt $\mathcal{I}$ mit $\mathcal{I}(\alpha) = 0$.
3. α tautologisch $:\Leftrightarrow$ Für alle $\mathcal{I}$ gilt $\mathcal{I}(\alpha) = 1$.
4. α widerspruchsvoll $:\Leftrightarrow$ Für alle $\mathcal{I}$ gilt $\mathcal{I}(\alpha) = 0$.

Beispiele:

	erfüllbar	falsifizierbar	tautologisch	widerspruchsvoll
$\alpha = A \wedge \neg A$	nein	ja	nein	ja
$\alpha = A \vee \neg A$				
$\alpha = A$				

Semantik der Aussagenlogik

Definition 8 (Erfüllbarkeitsbegriffe)

Sei α eine aussagenlogische Formel. Dann seien folgende Erfüllbarkeitsbegriffe vereinbart.

1. α erfüllbar $:\Leftrightarrow$ Es gibt $\mathcal{I}$ mit $\mathcal{I}(\alpha) = 1$.
2. α falsifizierbar $:\Leftrightarrow$ Es gibt $\mathcal{I}$ mit $\mathcal{I}(\alpha) = 0$.
3. α tautologisch $:\Leftrightarrow$ Für alle $\mathcal{I}$ gilt $\mathcal{I}(\alpha) = 1$.
4. α widerspruchsvoll $:\Leftrightarrow$ Für alle $\mathcal{I}$ gilt $\mathcal{I}(\alpha) = 0$.

Beispiele:

	erfüllbar	falsifizierbar	tautologisch	widerspruchsvoll
$\alpha = A \wedge \neg A$	nein	ja	nein	ja
$\alpha = A \vee \neg A$	ja	nein	ja	nein
$\alpha = A$				

Semantik der Aussagenlogik

Definition 8 (Erfüllbarkeitsbegriffe)

Sei α eine aussagenlogische Formel. Dann seien folgende Erfüllbarkeitsbegriffe vereinbart.

1. α erfüllbar $:\Leftrightarrow$ Es gibt $\mathcal{I}$ mit $\mathcal{I}(\alpha) = 1$.
2. α falsifizierbar $:\Leftrightarrow$ Es gibt $\mathcal{I}$ mit $\mathcal{I}(\alpha) = 0$.
3. α tautologisch $:\Leftrightarrow$ Für alle $\mathcal{I}$ gilt $\mathcal{I}(\alpha) = 1$.
4. α widerspruchsvoll $:\Leftrightarrow$ Für alle $\mathcal{I}$ gilt $\mathcal{I}(\alpha) = 0$.

Beispiele:

	erfüllbar	falsifizierbar	tautologisch	widerspruchsvoll
$\alpha = A \wedge \neg A$	nein	ja	nein	ja
$\alpha = A \vee \neg A$	ja	nein	ja	nein
$\alpha = A$	ja	ja	nein	nein

Bemerkungen:

- Die Eigenschaften gehören jeweils paarweise zusammen: die Erfüllbarkeit ist die Verneinung der Widersprüchlichkeit, die Falsifizierbarkeit ist die Verneinung der Tautologie-Eigenschaft.
- Für jede aussagenlogische (und auch prädikatenlogische) Formel gelten genau zwei der gerade definierten Eigenschaften *erfüllbar, falsifizierbar, tautologisch* und *widerspruchsvoll*.
- Es gibt dafür wie im vorstehenden Beispiel genau drei Möglichkeiten. Die vierte Möglichkeit, dass eine Formel gleichzeitig tautologisch und widerspruchsvoll sein kann (d.h. damit auch nicht erfüllbar und nicht falsifizierbar) tritt nicht auf, da jede Formel nur einen der beiden Wahrheitswerte haben kann (Satz vom ausgeschlossenen Widerspruch).

Semantik der Aussagenlogik

Satz 2 (Erfüllbarkeitsbegriffe)

Für eine aussagenlogische Formel α gilt:

	α	widerspruchsvoll
$\Leftrightarrow$	$\neg\alpha$	tautologisch
$\Leftrightarrow$	α	nicht erfüllbar
$\Leftrightarrow$	$\neg\alpha$	nicht falsifizierbar

Semantik der Aussagenlogik

Definition 9 (Semantische Folgerung)

Seien α und β aussagenlogische Formeln. β ist semantische Folgerung aus α

$$\alpha \models \beta \quad :\Leftrightarrow \quad \begin{array}{l} \text{Für alle Bewertungen } \mathcal{I} \text{ gilt} \\ \text{mit } \mathcal{I}(\alpha) = 1 \text{ auch } \mathcal{I}(\beta) = 1, \text{ d. h.} \\ (\mathcal{I}(\alpha) = 1 \quad \Rightarrow \quad \mathcal{I}(\beta) = 1) \end{array}$$

Beispiel:

$\alpha = A \wedge (\neg A \vee B), \quad \beta = B$ Gilt $\alpha \models \beta$?

A	B	α	β	$\alpha \rightarrow \beta$
0	0			
0	1			
1	0			
1	1			

Semantik der Aussagenlogik

Definition 9 (Semantische Folgerung)

Seien α und β aussagenlogische Formeln. β ist semantische Folgerung aus α

$$\alpha \models \beta \quad :\Leftrightarrow \quad \begin{array}{l} \text{Für alle Bewertungen } \mathcal{I} \text{ gilt} \\ \text{mit } \mathcal{I}(\alpha) = 1 \text{ auch } \mathcal{I}(\beta) = 1, \text{ d. h.} \\ (\mathcal{I}(\alpha) = 1 \quad \Rightarrow \quad \mathcal{I}(\beta) = 1) \end{array}$$

Beispiel:

$\alpha = A \wedge (\neg A \vee B), \quad \beta = B$ Gilt $\alpha \models \beta$?

A	B	α	β	$\alpha \rightarrow \beta$
0	0	0	0	1
0	1	0	1	1
1	0	0	0	1
1	1	1	1	1

Semantik der Aussagenlogik

Zeichen der Objektsprache

$\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ sind Junktoren. Ein Junktor verbindet zwei Formeln zu einer neuen Formel.

Insbesondere *definiert* – im Sinne von berechnet – ein Junktor für eine gegebene Funktion $\mathcal{I}$ (Bewertung oder Interpretation genannt) den Wert der Funktion $\mathcal{I}$ der neuen Formel. Die Berechnung geschieht auf Grundlage der $\mathcal{I}$-Werte derjenigen Formeln, zwischen denen der Junktor steht.

Semantik der Aussagenlogik

Zeichen der Metasprache

$\models$ Semantisches (Formel-)Folgerungszeichen. Die rechte Seite folgt aus der linken; beide Seiten sind Formeln.

$\Rightarrow$ Semantisches (Aussagen-)Folgerungszeichen. Die rechte Seite folgt aus der linken; die Seiten müssen Aussagen sein.

Beachte, dass der Begriff „Folgerung“ sowohl für die rechte Seite als auch für den gesamten Ausdruck verwendet wird. Eine Folgerung – als gesamter Ausdruck gesehen – macht eine Aussage bzw. stellt eine Behauptung auf hinsichtlich *aller* Interpretationen der rechten und der linken Seite.
D. h., das $\models$-Zeichen stellt eine Behauptung hinsichtlich aller Funktionen $\mathcal{I}$ auf; das $\Rightarrow$-Zeichen stellt eine Behauptung hinsichtlich aller Interpretationen auf, die zwischen dem Schreiber und dem Leser der Folgerung vereinbart wurden. Stichwort: Pragmatik.

$\approx$ Semantisches (Formel-)Äquivalenzzeichen. Die rechte Seite folgt aus der linken, und die linke Seite folgt aus der rechten; beide Seiten sind Formeln.

$\Leftrightarrow$ Semantisches (Aussagen-)Äquivalenzzeichen. Die rechte Seite folgt aus der linken, und die linke Seite folgt aus der rechten; die Seiten müssen Aussagen sein.

$\rightsquigarrow$ Zeichen zum Zusammenfassen von Umformungs- und Auswertungsschritten. Darüberhinaus ist für dieses Zeichen keine spezielle Semantik festgelegt.

Bemerkungen:

- Formeln sind wohlgeformte Ausdrücke in der Objektsprache.
- Mit den Zeichen $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ und den Atomen werden in der Aussagenlogik Formeln konstruiert. Jede Formel hat für eine gegebene Bewertung $\mathcal{I}$ einen Wert, nämlich 0 oder 1.
- Mit den Zeichen $\models$, $\Rightarrow$, $\approx$ und $\Leftrightarrow$ werden Aussagen in der Metasprache aufgestellt. Diese sind keine aussagenlogischen Formeln. Von Aussagen (Behauptungen) kann festgestellt werden, ob sie wahr oder falsch sind. Die aufgeführten Zeichen dienen als Abkürzungen zum Aufschreiben dieser speziellen Aussagen.

Beispiele für Formeln aus Aussagenlogik und Algebra:

- $\alpha \wedge \beta$ (hat unter Bewertung $\mathcal{I}$ den Wahrheitswert $\mathcal{I}(\alpha \wedge \beta)$)
- $4 + 3$
- $\neg A$
- $\sqrt{7}$
- $\alpha \rightarrow \beta$

Beispiele für Aussagen (Aussageformen):

- „7 ist eine gerade Zahl“ (Diese Aussage ist falsch.)
- „x ist eine gerade Zahl“ (Aussage*form*)
- $4 + 3 = \sqrt{7}$
- $\alpha \models \beta$ (mit α und β als Namen für konkrete Formeln: Aussage)
- $\alpha \models \beta$ (mit α und β als Formelvariablen: Aussageform)
- „α ist erfüllbar“

Semantik der Aussagenlogik

Weitere Vereinbarungen zur Syntax:

1. $\rightarrow$ und $\leftrightarrow$ binden stärker als $\models$ und $\approx$

2. $\models$ bindet stärker als $\Rightarrow$, $\Leftrightarrow$ und $\rightsquigarrow$

3. $\models$ mit Mengenschreibweise: $\alpha_1, \ldots, \alpha_n \models \beta$ bzw. $M \models \beta$.

   Die Formeln einer Menge werden als mit „$\wedge$“ verknüpft angesehen.
   $\emptyset \models \beta$ wird abgekürzt geschrieben als $\models \beta$

Bemerkungen:

- Offensichtlich gilt: $\emptyset \models \beta \quad \Leftrightarrow \quad \beta$ tautologisch
  β ist unter allen Interpretationen wahr, unter denen die leere Menge von Formeln wahr ist. Die leere Menge von Formeln ist unter allen möglichen Interpretationen wahr. Also muss β unter allen möglichen Interpretationen wahr sein. Also ist β eine Tautologie. – Formal:

- Was bedeutet $M \models \beta$?
  $M \models \beta \quad \Leftrightarrow \quad \forall \mathcal{I} : (\mathcal{I}(M) = 1 \quad \Rightarrow \quad \mathcal{I}(\beta) = 1)$

- Wann ist $\mathcal{I}(M) = 1$?
  $\mathcal{I}(M) = 1 \quad \Leftrightarrow \quad \forall \alpha \in M : \mathcal{I}(\alpha) = 1$ genauer: $\forall \alpha : (\alpha \in M \quad \Rightarrow \quad \mathcal{I}(\alpha) = 1)$
  $\mathcal{I}(M) = 1$ gilt also genau dann, wenn $\forall \alpha : (\alpha \in M \quad \Rightarrow \quad \mathcal{I}(\alpha) = 1)$ wahr ist.

- Mit $M = \emptyset$ lässt sich aus $\alpha \in M$ alles folgern („ex falso sequitur quod libet“), also auch $\mathcal{I}(\alpha) = 1$, unabhängig von $\mathcal{I}$. Also ist $\mathcal{I}(M) = 1$ für jedes $\mathcal{I}$ wahr.
  Damit $\mathcal{I}(M) = 1 \quad \Rightarrow \quad \mathcal{I}(\beta) = 1$ eine Folgerung bleibt, muss auch $\mathcal{I}(\beta) = 1$ für jedes $\mathcal{I}$ wahr sein. Also ist β tautologisch.

Eigenschaften des Folgerungsbegriffs

Lemma 1 (über aussagenlogische Formeln α und β)

1. $\alpha \models \beta \quad \Leftrightarrow \quad \alpha \rightarrow \beta$ tautologisch
2. $\alpha \models \beta \quad \Leftrightarrow \quad \alpha \leftrightarrow (\alpha \wedge \beta)$ tautologisch
3. $\alpha \models \beta \quad \Leftrightarrow \quad \alpha \wedge \neg\beta$ widerspruchsvoll
4. α widerspruchsvoll $\quad \Leftrightarrow \quad$ Für alle β gilt $\alpha \models \beta$ „Ex falso quod libet.“
5. α widerspruchsvoll $\quad \Leftrightarrow \quad$ Es gibt γ mit $\alpha \models (\gamma \wedge \neg\gamma)$

Eigenschaften des Folgerungsbegriffs

Lemma 2 (Deduktionstheorem)

Seien α, β aussagenlogische Formeln, M eine Menge aussagenlogischer Formeln. Dann gilt:

$$M \cup \{\alpha\} \models \beta \quad \Leftrightarrow \quad M \models \alpha \rightarrow \beta$$

Bemerkungen:

- Das Deduktionstheorem macht Aussagen über die Folgerbarkeit von Implikationen (Regeln).

$$M \models \underbrace{\alpha \rightarrow \beta}_{\text{Regel}}$$

  Diese Regeln beschreiben für M allgemeingültige Zusammenhänge. M kann als die Modellierung eines Ausschnitts unserer Welt aufgefasst werden: Wenn die Modellierung von M zutrifft (wahr ist, $\mathcal{I}(M) = 1$), so ist auch die Regel $\alpha \rightarrow \beta$ wahr ($\mathcal{I}(\alpha \rightarrow \beta) = 1$).
- Das Deduktionstheorem zeigt, welche Regeln in Bezug auf ein Modell gültig sind.

Eigenschaften des Folgerungsbegriffs

Beweis 1 (Deduktionstheorem)

Sei M eine Menge von Formeln und sei $\mathcal{I}$ eine Bewertung mit $\mathcal{I}(M) = 1$.

1. Fall. $\mathcal{I}(\alpha) = 0,\ \mathcal{I}(\beta) = 0 \quad \Rightarrow \quad \mathcal{I}(M \cup \{\alpha\}) = 0,\ \mathcal{I}(\alpha \to \beta) = 1$
2. Fall. $\mathcal{I}(\alpha) = 0,\ \mathcal{I}(\beta) = 1$ analog
3. Fall. $\mathcal{I}(\alpha) = 1,\ \mathcal{I}(\beta) = 0 \quad \Rightarrow \quad \mathcal{I}(M \cup \{\alpha\}) = 1,\ \mathcal{I}(\alpha \to \beta) = 0$
   $\Rightarrow \quad M \cup \{\alpha\} \not\models \beta,\ M \not\models (\alpha \to \beta)$
4. Fall. $\mathcal{I}(\alpha) = 1,\ \mathcal{I}(\beta) = 1 \quad \Rightarrow \quad \mathcal{I}(M \cup \{\alpha\}) = 1,\ \mathcal{I}(\alpha \to \beta) = 1$

Tritt bei Betrachtung aller Bewertungen Fall 3 niemals auf, so gilt:

$$M \cup \{\alpha\} \models \beta \text{ und } M \models (\alpha \to \beta)$$

Tritt bei einer Bewertung Fall 3 auf: $M \cup \{\alpha\} \not\models \beta$ und $M \not\models (\alpha \to \beta)$

Also gilt insgesamt: $M \cup \{\alpha\} \models \beta \quad \Leftrightarrow \quad M \models (\alpha \to \beta)$ □

Bemerkungen:

- Die Art der Definition von Bewertungen erlaubt sofort auch die Bewertung von (unendlichen) Mengen von Formeln. Die Begriffe „erfüllbar“, „falsifizierbar“, „tautologisch“ und „widerspruchsvoll“ lassen sich dementsprechend auf Formelmengen ausdehnen.

Eigenschaften des Folgerungsbegriffs

Korollar 1 (Deduktionstheorem für endliches M)
Seien α, β aussagenlogische Formeln, M eine endliche Menge aussagenlogischer Formeln. Dann gilt:

$$M \models \beta \quad \Leftrightarrow \quad \models M \rightarrow \beta$$

Lemma 3 (Interpolationstheorem)
Seien α, β aussagenlogische Formeln, und α nicht widerspruchsvoll, β keine Tautologie. Dann gilt:

$\alpha \models \beta \quad \Leftrightarrow$ Es gibt eine Formel γ mit
atoms$(\gamma) \subseteq$ *atoms*$(\alpha) \cap$ *atoms*(β) und
$\alpha \models \gamma$ und $\gamma \models \beta$.

Bemerkungen zum Interpolationstheorem:

- Abgesehen von den trivialen Fällen (α widerspruchsvoll oder β tautologisch) sind nur Formelbestandteile, die in Prämisse und Konklusion gemeinsam auftreten, verantworlich für die Gültigkeit der Folgerungsrelation.

- Beispiel:
  $A \wedge B \models B \vee C$
  $\Rightarrow$ Mit Interpolante B gilt $A \wedge B \models B$ und $B \models B \vee C$

Äquivalenz

Problematik:

Experte 1 —— Formalisierung 1 ——> α_1

Experte 2 —— Formalisierung 2 ——> α_2

Es soll gelten (bei Absprache der Bezeichner):

- *atoms*$(\alpha_1) =$ *atoms*(α_2)
- zumindest sei *atoms*$(\alpha_1) \cap$ *atoms*(α_2) „groß“

Jedoch: $\alpha_1 \neq \alpha_2$

Frage: Wann sind logische Formeln äquivalent?

- Bei Gleichheit der Zeichenketten?
- Bei Gleichheit der Zeichenketten modulo Klammerung / Kommutativität?
- Bei Gleicheit von *atoms*(α_1) und *atoms*(α_2)?
- Bei Rückgabe gleicher Antworten auf gleiche Fragen?
- . . . ?

Äquivalenz

Definition 10 (logische Äquivalenz)

Zwei Formeln α und β heißen logisch äquivalent, $\alpha \approx \beta$, genau dann, wenn für alle Interpretationen $\mathcal{I}$ gilt:

$$\mathcal{I}(\alpha) = \mathcal{I}(\beta)$$

Hiermit folgt:

a) $\alpha \approx \beta \quad \Leftrightarrow \quad \alpha \leftrightarrow \beta$ tautologisch

b) α, β widerspruchsvoll $\quad \Rightarrow \quad \alpha \approx \beta$

c) α, β tautologisch $\quad \Rightarrow \quad \alpha \approx \beta$

d) $\alpha \models \beta \quad \Leftrightarrow \quad \alpha \approx \alpha \wedge \beta$

Äquivalenz

Lemma 4

Sei α eine aussagenlogische Formel, γ eine Teilformel von α und δ eine Formel mit $\gamma \approx \delta$.

Weiterhin sei β_1 das Ergebnis der Ersetzung *eines* Vorkommens von γ in α durch δ. Dann gilt:

$$\alpha \approx \beta_1$$

Sei β_* das Ergebnis der Ersetzung *aller* Vorkommen von γ in α durch δ. Dann gilt:

$$\alpha \approx \beta_*$$

Äquivalenz

Wichtige Äquivalenzen zur Formeltransformation / -vereinfachung

Junktoren

$$\alpha \to \beta \approx \neg\alpha \vee \beta$$
$$\alpha \leftrightarrow \beta \approx (\alpha \to \beta) \wedge (\beta \to \alpha)$$
$$\approx (\alpha \wedge \beta) \vee (\neg\beta \wedge \neg\alpha)$$

Vererbung

$$\alpha \approx \beta \quad \Rightarrow \quad \neg\alpha \approx \neg\beta$$
$$\alpha \approx \beta \quad \Rightarrow \quad \gamma \wedge \alpha \approx \gamma \wedge \beta$$
$$\alpha \approx \beta \quad \Rightarrow \quad \gamma \vee \alpha \approx \gamma \vee \beta$$
$$\alpha \approx \beta \quad \Rightarrow \quad \gamma \to \alpha \approx \gamma \to \beta$$
$$\alpha \approx \beta \quad \Rightarrow \quad \alpha \to \gamma \approx \beta \to \gamma$$
$$\alpha \approx \beta \quad \Rightarrow \quad \alpha \leftrightarrow \gamma \approx \beta \leftrightarrow \gamma$$

Negation als Involution

$$\neg\neg\alpha \approx \alpha$$

Idempotenz

$$\alpha \vee \alpha \approx \alpha$$
$$\alpha \wedge \alpha \approx \alpha$$

Äquivalenz

Wichtige Äquivalenzen zur Formeltransformation / -vereinfachung (Fortsetzung)

Kommutativität	$\alpha \vee \beta \approx \beta \vee \alpha$
	$\alpha \wedge \beta \approx \beta \wedge \alpha$
	$\alpha \leftrightarrow \beta \approx \beta \leftrightarrow \alpha$
Assoziativität	$(\alpha \vee \beta) \vee \gamma \approx \alpha \vee (\beta \vee \gamma)$
	$(\alpha \wedge \beta) \wedge \gamma \approx \alpha \wedge (\beta \wedge \gamma)$
Distributivität	$(\alpha \wedge \beta) \vee \gamma \approx (\alpha \vee \gamma) \wedge (\beta \vee \gamma)$
	$(\alpha \vee \beta) \wedge \gamma \approx (\alpha \wedge \gamma) \vee (\beta \wedge \gamma)$
De Morgan	$\neg(\alpha \wedge \beta) \approx \neg\alpha \vee \neg\beta$
	$\neg(\alpha \vee \beta) \approx \neg\alpha \wedge \neg\beta$

Äquivalenz

Definition 11 (Erfüllbarkeitsäquivalenz)

Zwei Formeln α und β heißen erfüllbarkeitsäquivalent, $\alpha \approx_{\text{sat}} \beta$, genau dann, wenn gilt:

$$\alpha \text{ erfüllbar} \quad \Leftrightarrow \quad \beta \text{ erfüllbar}$$

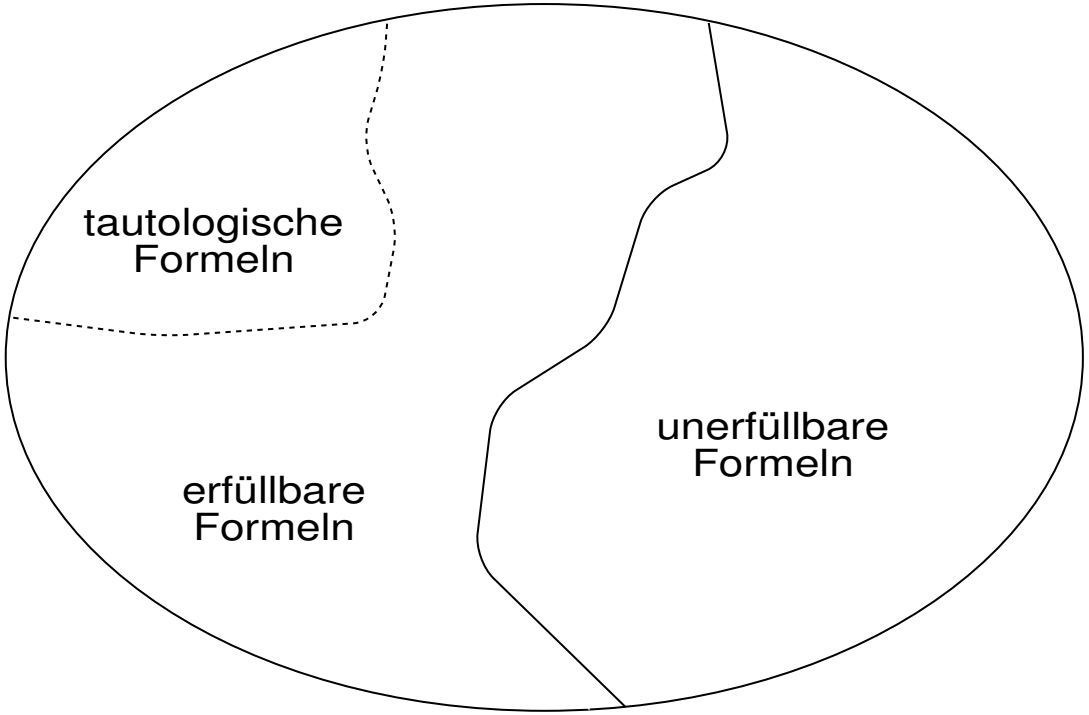
Unterschied zur logischen Äquivalenz:

$$\alpha \approx \beta \quad \Leftrightarrow \quad \forall \mathcal{I}: \quad \mathcal{I}(\alpha) = \mathcal{I}(\beta)$$

$$\alpha \approx_{\text{sat}} \beta \quad \Leftrightarrow \quad (\exists \mathcal{I}: \mathcal{I}(\alpha) = 1 \quad \Leftrightarrow \quad \exists \mathcal{I}: \mathcal{I}(\beta) = 1)$$

Äquivalenz

Veranschaulichung der Formeln hinsichtlich ihrer Erfüllbarkeit:



Bemerkungen:

- Für die Erfüllbarkeitsäquivalenz gilt:
  a) $\alpha \approx \beta \quad \Rightarrow \quad \alpha \approx_{\text{sat}} \beta$
  b) α, β widerspruchsvoll $\quad \Rightarrow \quad \alpha \approx_{\text{sat}} \beta$
  c) α, β erfüllbar $\quad \Rightarrow \quad \alpha \approx_{\text{sat}} \beta$

- Beachte, es gibt keine „Vererbung“ der Erfüllbarkeitsäquivalenz: Es gibt Formeln $\alpha, \beta, \gamma, \delta$ mit
  - γ ist Teilformel in α
  - β_1 ist das Ergebnis der Ersetzung eines Vorkommens von γ in α durch δ.
  - $\gamma \approx_{\text{sat}} \delta$ aber $\alpha \not\approx_{\text{sat}} \beta_1$

- Beispiel:
  $\alpha = A \wedge \neg B, \quad \gamma = A, \quad \delta = B$
  $\rightsquigarrow \beta_1 = B \wedge \neg B$

  $A \approx_{\text{sat}} B$ aber $A \wedge \neg B \not\approx_{\text{sat}} B \wedge \neg B$

Kapitel L: II (Fortsetzung)

II. Aussagenlogik

- Syntax der Aussagenlogik
- Semantik der Aussagenlogik
- Eigenschaften des Folgerungsbegriffs
- Äquivalenz
- Formeltransformation
- Normalformen

- Bedeutung der Folgerung
- Erfüllbarkeitsalgorithmen
- Semantische Bäume
- Weiterentwicklung semantischer Bäume
- Syntaktische Schlussfolgerungsverfahren
- Erfüllbarkeitsprobleme

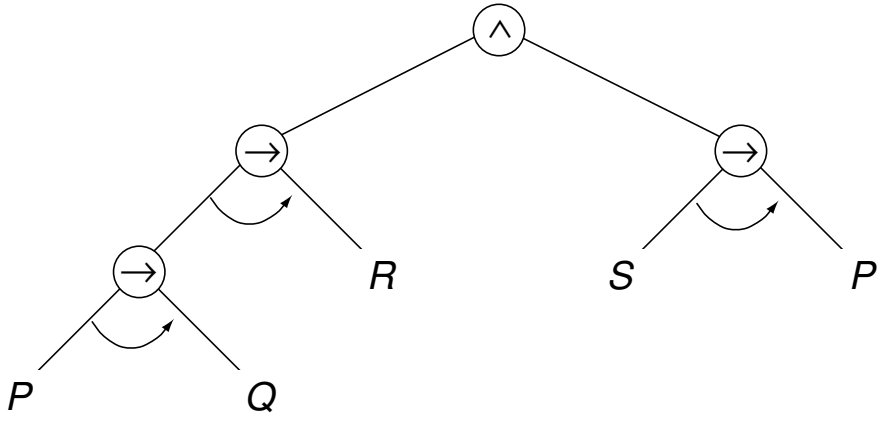
Formeltransformation

Jede Formel kann als Datenstruktur in der Form eines Baumes interpretiert werden:

- Junktoren entsprechen den inneren Knoten
- Atome entsprechen den Blättern
- (Teil-)Formeln entsprechen (Teil-)Bäumen
- Bei nicht-assoziativen Junktoren ist die Reihenfolge der Nachfolger eines Knotens zu beachten.

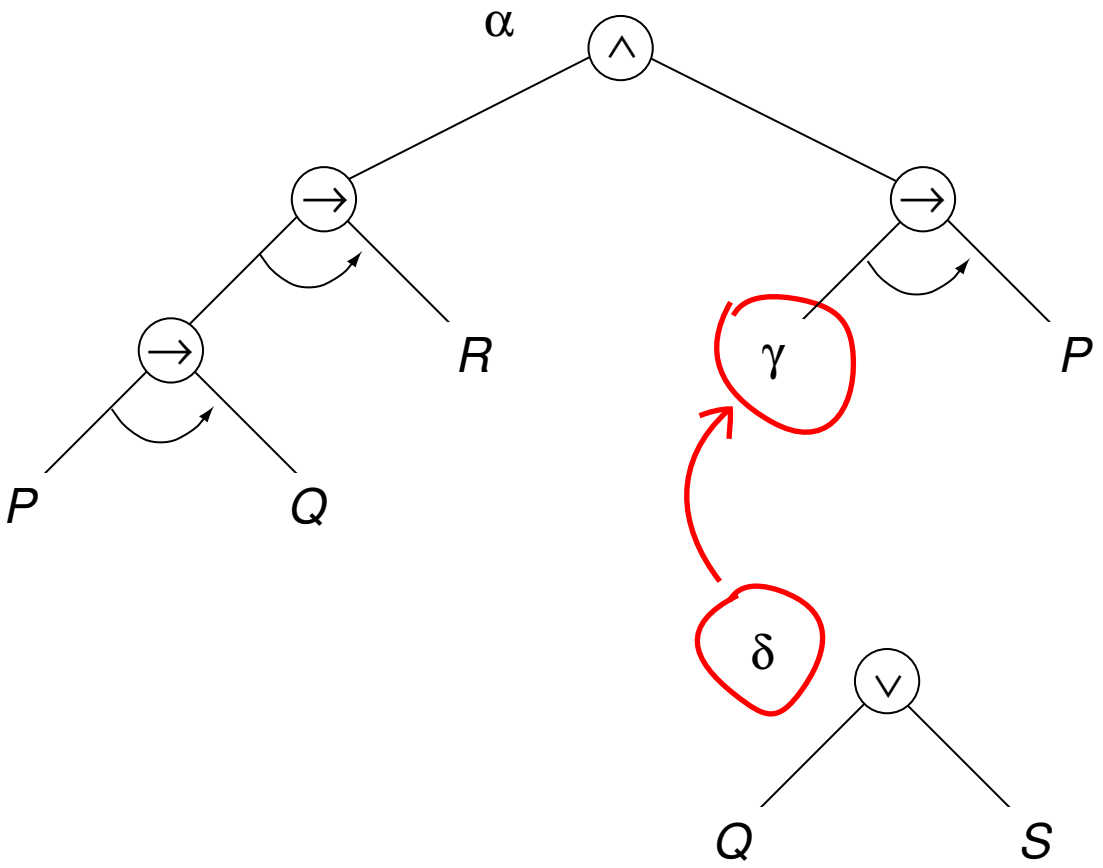
Beispiel:

$\alpha = ((P \rightarrow Q) \rightarrow R) \wedge (S \rightarrow P)$



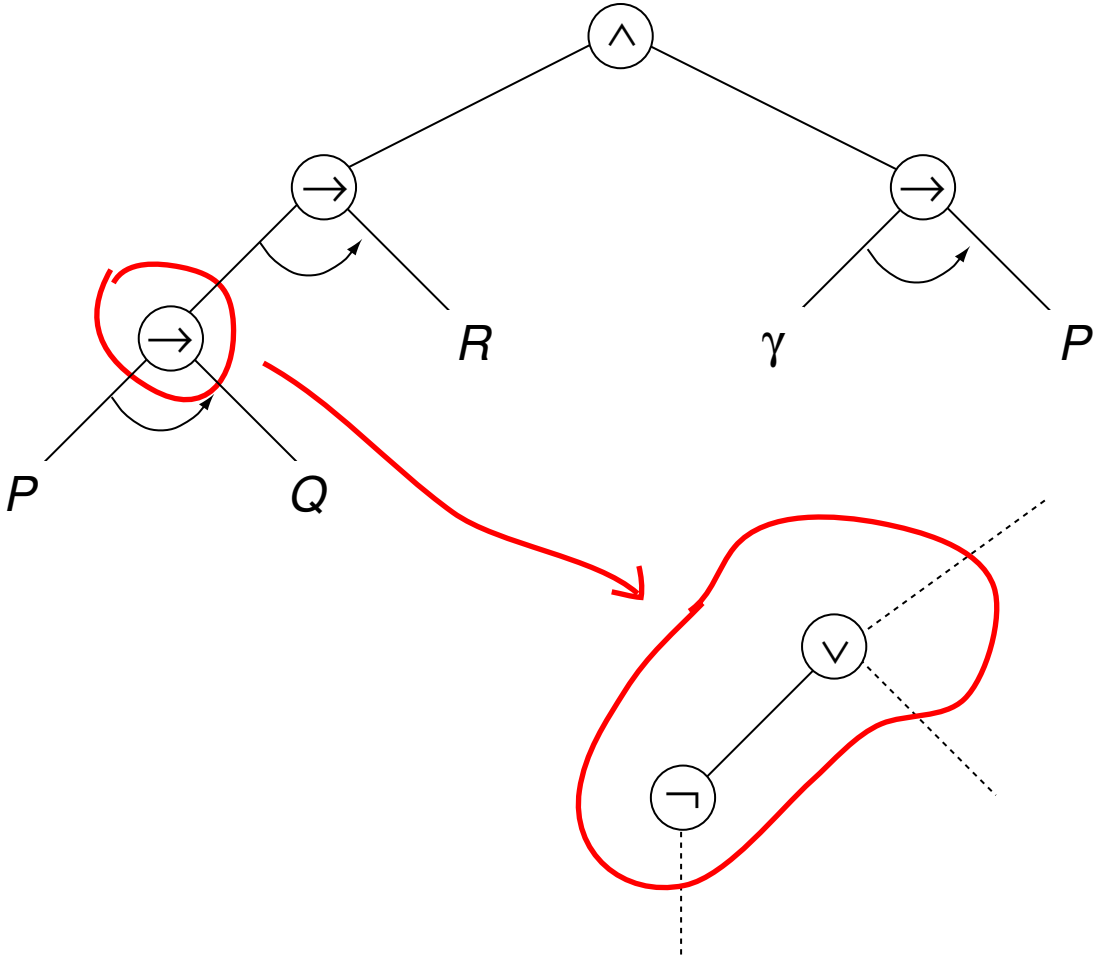
Formeltransformation

Die Ersetzung eines Vorkommens von γ in α durch δ entspricht der Ersetzung eines Blattes (oder Teilbaums) durch einen anderen Baum.



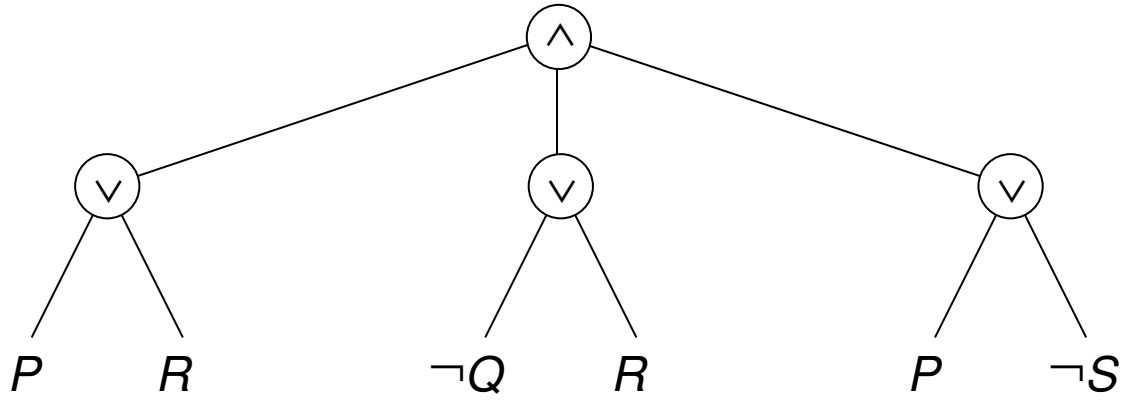
Formeltransformation

Aus Sicht der maschinellen Verarbeitung hätte man gerne kanonische Formeln bzw. Bäume:



Formeltransformation

Aus Sicht der maschinellen Verarbeitung hätte man gerne kanonische Formeln bzw. Bäume:



Bemerkung: $\wedge$ und $\vee$ können als n-äre Knoten aufgefasst werden.

Fragen:

- Was an Kanonisierung ist möglich?
  (unter den verschiedenen Äquivalenzbegriffen)
- Wie operationalisiert (Algorithmus) man Kanonisierung?

Normalformen

Erste Stufe einer Normalisierung:

- Reduzierung der Junktorenmenge.
- Ersetzung von $\rightarrow$, $\leftrightarrow$ entsprechend den Äquivalenzen.

„$\rightarrow$“-Ersetzung: Länge der entstehenden Formel ist . . .

„$\leftrightarrow$“-Ersetzung: Länge der entstehenden Formel ist . . .

Normalformen

Erste Stufe einer Normalisierung:

- Reduzierung der Junktorenmenge.
- Ersetzung von $\rightarrow$, $\leftrightarrow$ entsprechend den Äquivalenzen.

„$\rightarrow$“-Ersetzung: Länge der entstehenden Formel ist . . . linear in der Ausgangslänge

„$\leftrightarrow$“-Ersetzung: Länge der entstehenden Formel ist . . . exponentiell in der Ausgangslänge

Normalformen

Definition 13 (Negationsnormalform)

α ist in Negationsnormalform (NNF) genau dann, wenn jedes Negationszeichen in α unmittelbar vor einem Atom steht und α weder den Junktor $\rightarrow$ noch den Junktor $\leftrightarrow$ enthält:

1. Jede Primformel $\alpha = A,\ A \in \Sigma$ ist in NNF.
2. Jede negierte Primformel $\alpha = \neg A,\ A \in \Sigma$ ist in NNF.
3. Sind α, β in NNF, so sind es auch $(\alpha \wedge \beta)$ und $(\alpha \vee \beta)$.

Lemma 5

Zu jeder Formel α gibt es eine logisch äquivalente Formel β in NNF.

Beweisidee

Induktion mit Anwendung folgender Regeln:

Negation	$\neg\neg\alpha \approx \alpha$
De Morgan	$\neg(\alpha \wedge \beta) \approx \neg\alpha \vee \neg\beta$
	$\neg(\alpha \vee \beta) \approx \neg\alpha \wedge \neg\beta$

Normalformen

Algorithmus: NNF

Input: α. A formula in tree representation w/o $\rightarrow$ and $\leftrightarrow$.
neg. A flag which is initially 'FALSE'.

Output: A formula equivalent to α in NNF.

```
NNF (α, neg)
  IF α = ¬β
  THEN
    IF neg
    THEN RETURN(NNF(β, 'FALSE'))
    ELSE RETURN(NNF(β, 'TRUE'))
  ELSE_IF α = β ∧ γ
    IF neg
    THEN RETURN(NNF(β, 'TRUE') ∨ NNF(γ, 'TRUE'))
    ELSE RETURN(NNF(β, 'FALSE') ∧ NNF(γ, 'FALSE'))
  ELSE_IF α = β ∨ γ
    IF neg
    THEN RETURN(NNF(β, 'TRUE') ∧ NNF(γ, 'TRUE'))
    ELSE RETURN(NNF(β, 'FALSE') ∨ NNF(γ, 'FALSE'))
  ELSE   // alpha is atom.
    IF neg
    THEN RETURN(¬α)
    ELSE RETURN(α)
```

Frage: Wie ist die Laufzeit von NNF im $\mathcal{O}$-Kalkül?

Normalformen

Definition 14 (Literal)

Sei $A \in \Sigma$. A, $\neg A$ werden als Literale bezeichnet. Insbesondere heißt A positives Literal und $\neg A$ negatives Literal.

Definition 15 (Klausel)

Seien $L_1, \ldots, L_n$ Literale. Dann heißt $\alpha = L_1 \vee \ldots \vee L_n$ Klausel. Insbesondere heißt α

1. positive Klausel, falls $L_1, \ldots, L_n$ positive Literale sind,
2. negative Klausel, falls $L_1, \ldots, L_n$ negative Literale sind,
3. gemischte Klausel, falls nicht 1. und nicht 2. gilt,
4. Unit-Klausel, falls $n = 1$,
5. k-Klausel, falls $n \leq k$,
6. Krom-Klausel, falls $n \leq 2$,
7. Horn-Klausel, falls maximal ein Literal positiv ist,
8. definite Horn-Klausel, falls genau ein Literal positiv ist.

Normalformen

Definition 16 (Konjunktive Normalform)

Seien $\alpha_1, \ldots, \alpha_n$ Klauseln. Dann heißt $\alpha = \alpha_1 \wedge \ldots \wedge \alpha_n$ Formel in konjunktiver Normalform.

$$\text{KNF} = \{\alpha \mid \alpha \text{ ist in konjunktiver Normalform}\}$$

Weiterhin seien folgende Formelklassen vereinbart:

1. k-KNF. Alle Klauseln sind k-Klauseln (k beliebig aber fest).
2. HORN. Alle Klauseln sind Horn-Klauseln.
3. DHORN. Alle Klauseln sind definite Horn-Klauseln.

Normalformen

Lemma 6 (logisch äquivalente KNF)

Zu jeder Formel α gibt es eine logisch äquivalente Formel $\beta \in$ KNF – in Worten: „β in KNF“.

Beweisidee

1. Transformation in NNF.
2. Induktion mit Anwendung des Distributiv-Gesetzes:

$$(\alpha \wedge \beta) \vee \sigma \approx (\alpha \vee \sigma) \wedge (\beta \vee \sigma)$$

Normalformen

Algorithmus: EQ-CNF

Input: α. A formula in tree representation in NNF.

Output: A formula equivalent to α in CNF.

```
EQ-CNF(α)
  DO
    α_org = α
    IF  α = β ∧ γ
    THEN α = EQ-CNF(β) ∧ EQ-CNF(γ)

    ELSE_IF α = (δ ∧ ε) ∨ γ
      α = (δ ∨ γ) ∧ (ε ∨ γ)
    ELSE_IF α = β ∨ (δ ∧ ε)
      α = (β ∨ δ) ∧ (β ∨ ε)
    ELSE_IF α = β ∨ γ
      α = EQ-CNF(β) ∨ EQ-CNF(γ)
    ELSE   // α is literal; do nothing.

  WHILE (α_org ≠ α)
  RETURN(α)
```

Bemerkung: Laufzeit und Platzbedarf von EQ-CNF im $\mathcal{O}$-Kalkül sind exponentiell in $|\alpha|$.

Normalformen

Beispiel zum exponentiellen Platzbedarf.

Formel α in NNF:

$$\alpha = \bigvee_{1 \leq i \leq n} (A_{i,1} \wedge A_{i,2})$$

$\rightsquigarrow |\alpha| = 2n$

Normalformen

Beispiel zum exponentiellen Platzbedarf.

Formel α in NNF:

$$\alpha = \bigvee_{1 \leq i \leq n} (A_{i,1} \wedge A_{i,2})$$

$$\rightsquigarrow |\alpha| = 2n$$

KNF zu α:

$$\beta = \bigwedge_{j_1,\ldots,j_n \in \{1,2\}} (A_{1,j_1} \vee \ldots \vee A_{n,j_n})$$

$$\rightsquigarrow |\beta| = 2^n \cdot n$$

Normalformen

Beispiel zum exponentiellen Platzbedarf.

Formel α in NNF:

$$\alpha = \bigvee_{1 \leq i \leq n} (A_{i,1} \wedge A_{i,2})$$

$\rightsquigarrow |\alpha| = 2n$

KNF zu α:

$$\beta = \bigwedge_{j_1,\ldots,j_n \in \{1,2\}} (A_{1,j_1} \vee \ldots \vee A_{n,j_n})$$

$\rightsquigarrow |\beta| = 2^n \cdot n$

Konkret für $n = 3$:

$\alpha = (A_{1,1} \wedge A_{1,2}) \vee (A_{2,1} \wedge A_{2,2}) \vee (A_{3,1} \wedge A_{3,2})$ bzw.

$\alpha = (A_1 \wedge A_2) \vee (A_3 \wedge A_4) \vee (A_5 \wedge A_6)$

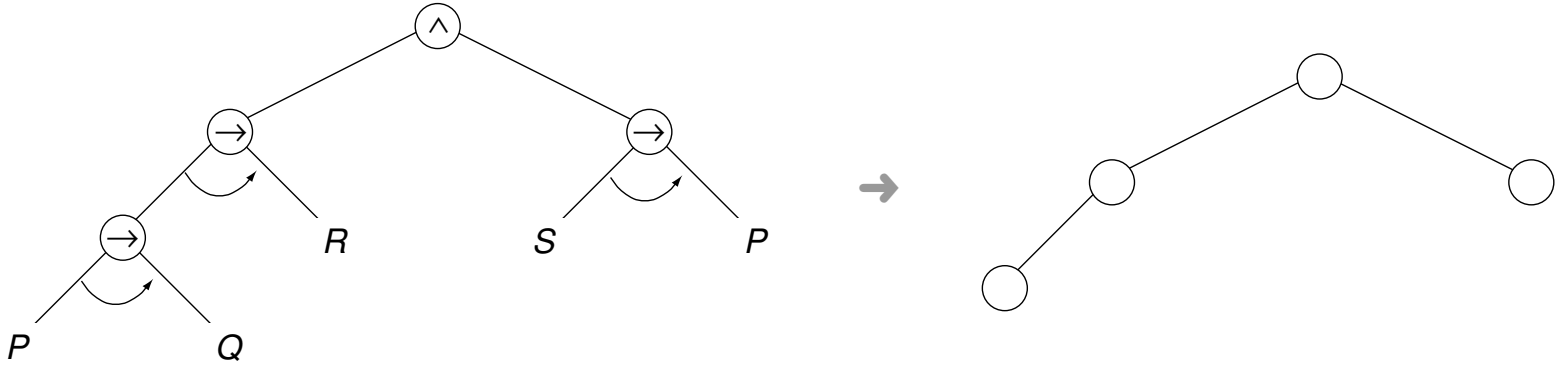
$\rightsquigarrow \beta = (A_{1,1} \vee A_{2,1} \vee A_{3,1}) \wedge \ldots \wedge (A_{1,2} \vee A_{2,2} \vee A_{3,2})$ mit $|\beta| = 8 \cdot 3$

Formeltransformation nach Tseitin

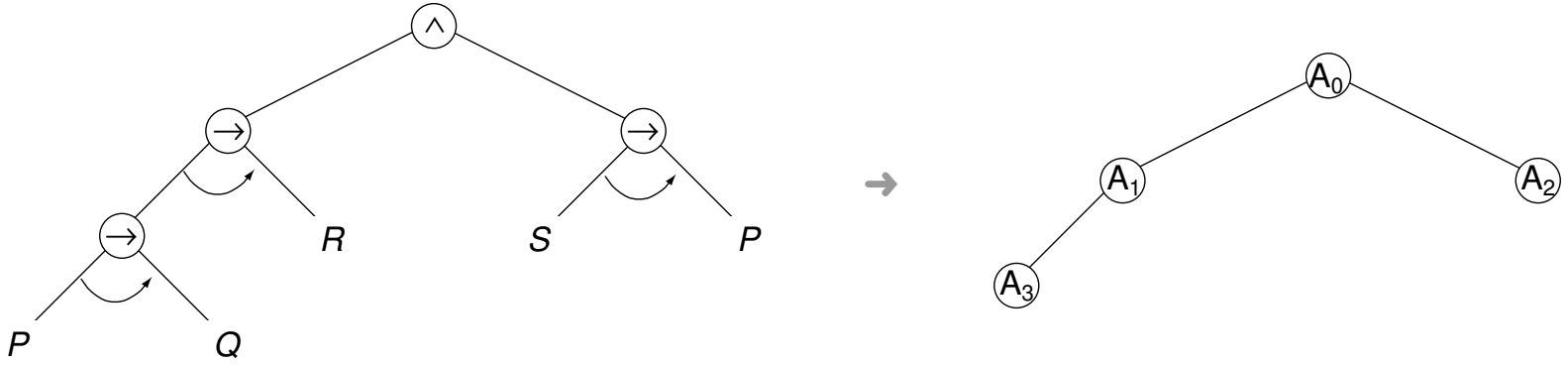
Andere Idee zur Erzeugung einer KNF aus α:

1. Beschreibung der *Formelstruktur* von α mit Hilfe einer neuen Formel, die erfüllbarkeitsäquivalent – aber nicht logisch äquivalent ist.
2. Umwandlung der neuen Formel in eine KNF in $\mathcal{O}(|\alpha|)$.

Beispiel: $((P \to Q) \to R) \wedge (S \to P)$



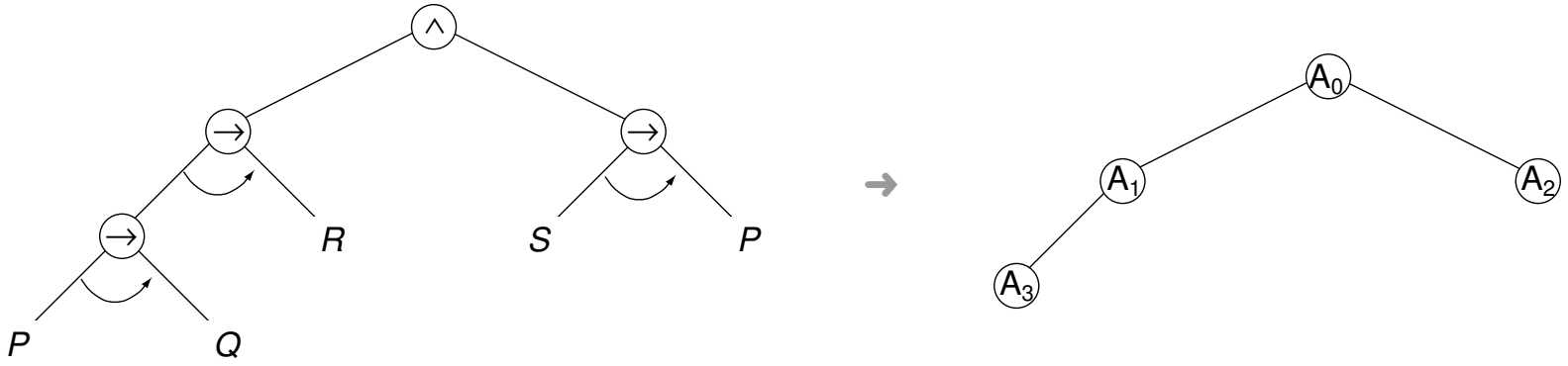
Formeltransformation nach Tseitin



Schritte:

1. Ersetzung der inneren Knoten durch neue Atome A_0, A_1, A_2, A_3.

Formeltransformation nach Tseitin

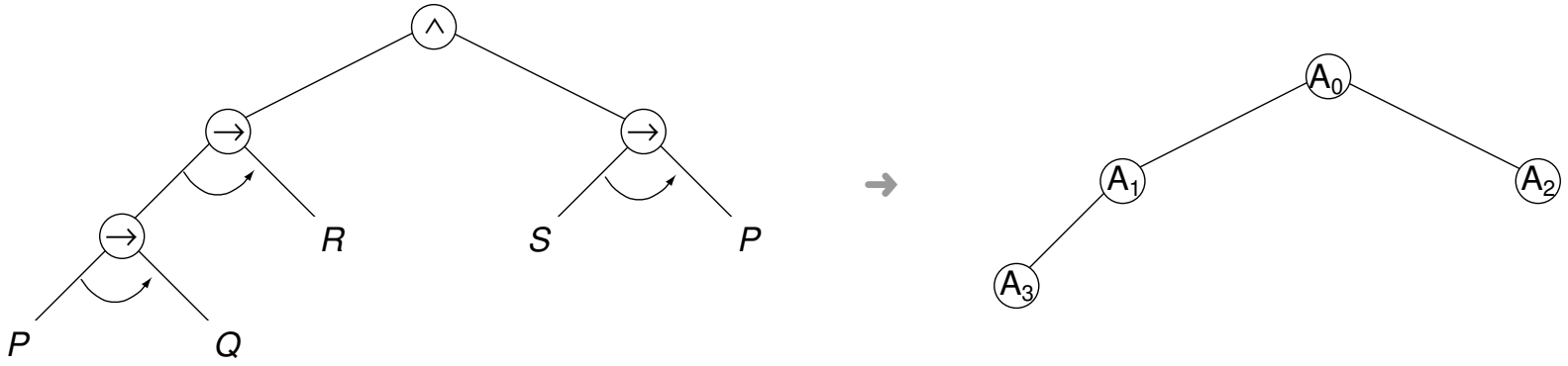


Schritte:

1. Ersetzung der inneren Knoten durch neue Atome A_0, A_1, A_2, A_3.
2. Einführung von Äquivalenzen für Zusammenhang:

$$\begin{aligned} A_0 &\leftrightarrow (A_1 \wedge A_2) \\ A_1 &\leftrightarrow (A_3 \rightarrow R) \\ A_2 &\leftrightarrow (S \rightarrow P) \\ A_3 &\leftrightarrow (P \rightarrow Q) \end{aligned}$$

Formeltransformation nach Tseitin



Schritte:

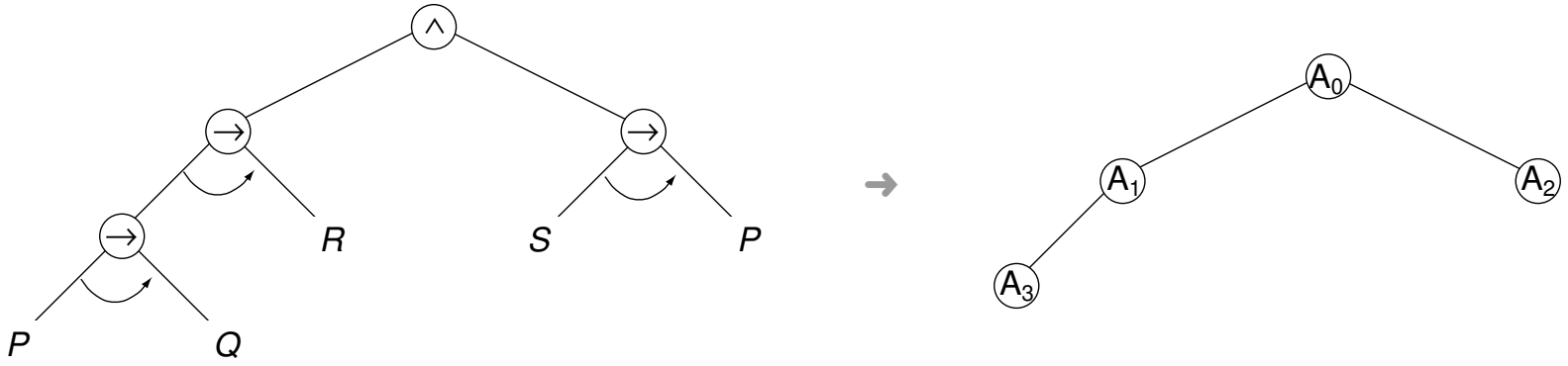
1. Ersetzung der inneren Knoten durch neue Atome A_0, A_1, A_2, A_3.
2. Einführung von Äquivalenzen für Zusammenhang:

$$\begin{aligned} A_0 &\leftrightarrow (A_1 \wedge A_2) \\ A_1 &\leftrightarrow (A_3 \rightarrow R) \\ A_2 &\leftrightarrow (S \rightarrow P) \\ A_3 &\leftrightarrow (P \rightarrow Q) \end{aligned}$$

3. Konjunktive Verknüpfung von Äquivalenzen und Gesamtformelrepräsentant:

$$(A_0 \leftrightarrow (A_1 \wedge A_2)) \quad \wedge \quad (A_1 \leftrightarrow (A_3 \rightarrow R)) \quad \wedge$$
$$(A_2 \leftrightarrow (S \rightarrow P)) \quad \wedge \quad (A_3 \leftrightarrow (P \rightarrow Q)) \quad \wedge \quad A_0$$

Formeltransformation nach Tseitin



Schritte:

1. Ersetzung der inneren Knoten durch neue Atome A_0, A_1, A_2, A_3.
2. Einführung von Äquivalenzen für Zusammenhang:
$$\begin{aligned} A_0 &\leftrightarrow (A_1 \wedge A_2) \\ A_1 &\leftrightarrow (A_3 \rightarrow R) \\ A_2 &\leftrightarrow (S \rightarrow P) \\ A_3 &\leftrightarrow (P \rightarrow Q) \end{aligned}$$
3. Konjunktive Verknüpfung von Äquivalenzen und Gesamtformelrepräsentant:
$$(A_0 \leftrightarrow (A_1 \wedge A_2)) \quad \wedge \quad (A_1 \leftrightarrow (A_3 \rightarrow R)) \quad \wedge$$
$$(A_2 \leftrightarrow (S \rightarrow P)) \quad \wedge \quad (A_3 \leftrightarrow (P \rightarrow Q)) \quad \wedge \quad A_0$$
4. Expansion (Transformation in KNF) der Äquivalenzen:
$$\alpha \leftrightarrow \beta \approx (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha) \approx (\neg\alpha \vee \beta) \wedge (\neg\beta \vee \alpha)$$

Bemerkungen:

- Wenn die Namenslänge der Atome durch eine Konstante beschränkt ist, so ist Schritt 4 (Expansion) pro Äquivalenz in konstanter Zeit und konstantem Platz möglich, da $|\beta| = 2$.

Formeltransformation nach Tseitin

Lemma 7 (erfüllbarkeitsäquivalente KNF)

Zu jeder aussagenlogischen Formel α gibt es eine erfüllbarkeitsäquivalente Formel $\beta \in$ KNF, wobei gilt:

1. $|\beta| \in \mathcal{O}(|\alpha|)$ Platz.

2. β ist in $\mathcal{O}(|\alpha|)$ Schritten aus der Baumdarstellung von α berechenbar. Generierung neuer Namen zählt als ein Schritt.

Bemerkungen:

- Das Lemma geht hinsichtlich der Namensgenerierung von einer konstanten Namenslänge $|A_i| = c$ aus. Jedoch – wenn n verschiedene Namen zu generieren sind, müßte genau genommen $|A_i| = \log(n)$ unterstellt werden. Mit $n \in \mathcal{O}(|\alpha|)$ würde die Berechnung von β also den Aufwand $\mathcal{O}(|\alpha| \cdot \log(|\alpha|))$ erfordern.
- Warum ist die im Lemma gemachte Vereinfachung bedenkenlos akzeptierbar?

Normalformen

Frage: Gegeben eine Formel $\in$ KNF. Inwieweit lassen sich die Klausellängen unter Beibehaltung der logischen Äquivalenz reduzieren?

Lemma 8 (Klausellänge $\approx$)

Für alle $k \geq 1$ gibt es ein $\alpha \in (k+1)$-KNF, so dass kein $\beta \in k$-KNF existiert mit

$$\alpha \approx \beta$$

Beweis 2

Wähle $\alpha = A_1 \vee \ldots \vee A_{k+1}$

$\beta = \beta_1 \wedge \ldots \wedge \beta_m$, sei $\beta \in k$-KNF; d. h., die β_i sind k-Klauseln

o.b.d.A. gelte: $A_{k+1} \notin \mathit{atoms}(\beta_1)$

Setze $\mathcal{I}$ so, dass $\mathcal{I}(\beta_1) = 0$ und $\mathcal{I}(A_{k+1}) = 1$

$\Rightarrow \mathcal{I}(\alpha) = 1 \neq 0 = \mathcal{I}(\beta)$

Normalformen

Frage: Gegeben eine Formel $\in$ KNF. Inwieweit lassen sich die Klausellängen unter Beibehaltung der Erfüllbarkeitsäquivalenz reduzieren?

Lemma 9 (Klausellänge $\approx_{\text{sat}}$)
Für jede Formel $\alpha \in$ KNF existiert eine erfüllbarkeitsäquivalente Formel $\beta \in 3$-KNF.

Beweisidee
Sei $\alpha_i = L_1 \vee \ldots \vee L_n$ eine Klausel aus α mit n Literalen, $n \geq 4$, und seien $A_0, \ldots, A_{n-4} \notin \mathit{atoms}(\alpha)$. Dann setze:

$$\begin{aligned}
\beta_0 &= L_1 \vee L_2 \vee A_0 \\
\beta_1 &= \neg A_0 \vee L_3 \vee A_1 \\
&\vdots \\
\beta_{n-4} &= \neg A_{n-5} \vee L_{n-2} \vee A_{n-4} \\
\beta_{n-3} &= \neg A_{n-4} \vee L_{n-1} \vee L_n
\end{aligned}$$

$\Rightarrow \quad \alpha_i \approx_{\text{sat}} \beta_0 \wedge \ldots \wedge \beta_{n-3}$

Bemerkungen:

- Keine Reduzierung auf Klausellängen unter 3 ist möglich bei einer Ausgangslänge der Klauseln von $k \geq 3$.

Normalformen

Definition 17 (disjunktive Normalform)

Sei $\alpha = \alpha_1 \vee \ldots \vee \alpha_n$ eine Disjunktion von Klauseln der Form $\alpha_i = L_{i,1} \wedge \ldots \wedge L_{i,m}$ mit den Literalen $L_{i,1}, \ldots, L_{i,m}$. Dann heißt α Formel in disjunktiver Normalform.

$$\text{DNF} = \{\alpha \mid \alpha \text{ ist in disjunktiver Normalform}\}$$

Lemma 10 (NNF einer Formel in KNF)

Sei $\alpha \in$ KNF, dann ist NNF($\neg\alpha$) $\in$ DNF.

Beweisidee

Induktion mit Anwendung folgender Regeln:

Negation	$\neg\neg\alpha \approx \alpha$
De Morgan	$\neg(\alpha \wedge \beta) \approx \neg\alpha \vee \neg\beta$
	$\neg(\alpha \vee \beta) \approx \neg\alpha \wedge \neg\beta$

Bemerkungen:

- Bei Formeln in disjunktiver Normalform ist der Klauselbegriff eher selten. Manche Autoren verwenden den Begriff „Monom“ in diesem Zusammenhang.
- Betrachten Sie die Normalformen KNF und DNF. Welche Normalform würden Sie wählen, um zu überprüfen, ob eine Formel tautologisch bzw. widerspruchsvoll ist?

Normalformen

Definition 18 (duale Formel)

Sei $\alpha \in$ NNF. Dann ist die duale Formel zu α, *dual*(α), wie folgt definiert:

1. *dual*$(A) := A$
2. *dual*$(\neg A) := \neg A$
3. *dual*$(\alpha \vee \beta) :=$ *dual*$(\alpha) \wedge$ *dual*(β)
4. *dual*$(\alpha \wedge \beta) :=$ *dual*$(\alpha) \vee$ *dual*(β)

Beispiel:

$\alpha = (A \wedge B) \vee (A \wedge (B \vee C))$

dual$(\alpha) = (A \vee B) \wedge (A \vee (B \wedge C))$

Normalformen

Lemma 11 (duale Formeln und Erfüllbarkeit)

Sei $\alpha \in$ KNF. Dann gilt:

1. *dual*$(\alpha) \in$ DNF
2. α tautologisch $\Leftrightarrow$ *dual*(α) widerspruchsvoll
3. α erfüllbar $\Leftrightarrow$ *dual*(α) falsifizierbar

Beweisidee

Sei $\mathcal{I}$ eine Bewertung.

Definiere: $\mathcal{I}_{dual}(A) := 1 - \mathcal{I}(A)$, für alle $A \in \Sigma$

Es folgt mit Induktion über den Aufbau von α in NNF (hier ohne Erläuterung):

$$\mathcal{I}_{dual}(\textbf{\textit{dual}}(\alpha)) = 1 - \mathcal{I}(\alpha)$$

Normalformen

Definition 19 (Mengenschreibweise für Formeln in KNF)

Für eine Klausel $L_1 \vee \ldots \vee L_m$ sei folgende Mengenschreibweise vereinbart

$$\{L_1, \ldots, L_m\}$$

Für eine Formel $\alpha_1 \wedge \ldots \wedge \alpha_n$ sei folgende Mengenschreibweise vereinbart

$$\{\alpha_1, \ldots, \alpha_n\}$$

Beispiel:

$\alpha = A \wedge (B \vee \neg C) \wedge (\neg A \vee C \vee D)$

$\rightsquigarrow \alpha = \{\{A\}, \{B, \neg C\}, \{\neg A, C, D\}\}$

Bemerkungen:

- Bei der Mengenschreibweise wird implizit eine Reduktion der Formel auf Basis folgender Äquivalenzen vorgenommen:
  1. $\alpha \approx \alpha \wedge \alpha$
  2. $\alpha \approx \alpha \vee \alpha$
  3. $\alpha \wedge \beta \approx \beta \wedge \alpha$
  4. $\alpha \vee \beta \approx \beta \vee \alpha$
  5. $(\alpha \wedge \beta) \wedge \gamma \approx \alpha \wedge (\beta \wedge \gamma)$
  6. $(\alpha \vee \beta) \vee \gamma \approx \alpha \vee (\beta \vee \gamma)$
- Mit Hilfe von Mengenoperatoren kann man Begiffe wie *Teilklausel* sehr einfach auf der Mengendarstellung definieren.
- In welcher Zeit ist die Transformation einer KNF als Menge durchführbar?

Kapitel L: II (Fortsetzung)

II. Aussagenlogik

- Syntax der Aussagenlogik
- Semantik der Aussagenlogik
- Eigenschaften des Folgerungsbegriffs
- Äquivalenz
- Formeltransformation
- Normalformen

- Bedeutung der Folgerung
- Erfüllbarkeitsalgorithmen
- Semantische Bäume
- Weiterentwicklung semantischer Bäume
- Syntaktische Schlussfolgerungsverfahren
- Erfüllbarkeitsprobleme

Bedeutung der Folgerung

Q: Warum ist der Begriff der Folgerung so wichtig?

A: Folgern (Deduktion) ist ein zentrales Konzept zum Arbeiten mit Modellen.

Hintergrund: Sei α ein Modell eines Weltausschnitts. Der Modellierer vereinbart mit den Verwendern des Modells α folgende Beziehung (Pragmatik):

„α ist erfüllt“ $\Leftrightarrow$ „Der Weltausschnitt wird durch α beschrieben.“

Sinnvolle Modelle entsprechen erfüllbaren Formeln. Sie dienen zur Simulation und werden konstruiert, um Vorhersagen zu machen.

Bedeutung der Folgerung

Q: Warum ist der Begriff der Folgerung so wichtig?

A: Folgern (Deduktion) ist ein zentrales Konzept zum Arbeiten mit Modellen.

Hintergrund: Sei α ein Modell eines Weltausschnitts. Der Modellierer vereinbart mit den Verwendern des Modells α folgende Beziehung (Pragmatik):

„α ist erfüllt“ $\Leftrightarrow$ „Der Weltausschnitt wird durch α beschrieben.“

Sinnvolle Modelle entsprechen erfüllbaren Formeln. Sie dienen zur Simulation und werden konstruiert, um Vorhersagen zu machen.

Sei β eine Folgerung aus α, in Zeichen: $\alpha \models \beta$, dann weiß man (aus der Definition der Folgerung):

$$\alpha \quad \approx \quad \alpha \wedge \beta$$

Das bedeutet aus Modellierungssicht:

- β ist immer wahr, wenn α wahr ist.
- β ist *verträglich* mit dem Modell α. β wird vom Modell α vorhergesagt.
- Die Folgerung hat den verträglichen Sachverhalt β *explizit* gemacht, sie hat ihn *bewiesen*.
- Beachte: Die Folgerung hat uns über die Veträglichkeit von β lediglich *informiert*. Auch ohne das Explizitmachen hätte $\alpha \models \beta$ gegolten.

Bedeutung der Folgerung

Q: Warum ist der Begriff der Folgerung so wichtig?

A: Folgern (Deduktion) ist ein zentrales Konzept zum Arbeiten mit Modellen.

Hintergrund: Sei α ein Modell eines Weltausschnitts. Der Modellierer vereinbart mit den Verwendern des Modells α folgende Beziehung (Pragmatik):

„α ist erfüllt“ $\Leftrightarrow$ „Der Weltausschnitt wird durch α beschrieben.“

Sinnvolle Modelle entsprechen erfüllbaren Formeln. Sie dienen zur Simulation und werden konstruiert, um Vorhersagen zu machen.

Sei β eine Folgerung aus α, in Zeichen: $\alpha \models \beta$, dann weiß man (aus der Definition der Folgerung):

$$\alpha \quad \approx \quad \alpha \wedge \beta$$

Das bedeutet aus Modellierungssicht:

- β ist immer wahr, wenn α wahr ist.
- β ist *verträglich* mit dem Modell α. β wird vom Modell α vorhergesagt.
- Die Folgerung hat den verträglichen Sachverhalt β *explizit* gemacht, sie hat ihn *bewiesen*.
- Beachte: Die Folgerung hat uns über die Veträglichkeit von β lediglich *informiert*. Auch ohne das Explizitmachen hätte $\alpha \models \beta$ gegolten.

Zusammengefasst. Arbeiten mit Modellen α heißt Folgerungen aus α zu erzeugen oder zu überprüfen, ob eine Formel β eine Folgerung aus α ist.

Bedeutung der Folgerung

Q: Wenn Folgern lediglich Explizitmachen ist, warum hat dann das Überprüfen oder Erzeugen von Folgerungen eine so große Bedeutung?

Bedeutung der Folgerung

Q: Wenn Folgern lediglich Explizitmachen ist, warum hat dann das Überprüfen oder Erzeugen von Folgerungen eine so große Bedeutung?

A: Hier ist ein Modell α. Gilt $\alpha \models$ „P3=5“ ?

α = (:AND P1=3 Q1=1 CYL2_IS_OK CYL1_IS_OK PIPE4_IS_OK PIPE3_IS_OK PIPE2_IS_OK PIPE1_IS_OK PUMP_IS_OK (:OR (:NOT PIPE1_IS_OK) (:AND P1=5 P2=5 Q1=2 Q2=2) (:AND P1=5 P2=5 Q1=1 Q2=1) (:AND P1=5 P2=5 Q1=0 Q2=0) (:AND P1=4 P2=4 Q1=2 Q2=2) (:AND P1=4 P2=4 Q1=1 Q2=1) (:AND P1=4 P2=4 Q1=0 Q2=0) (:AND P1=3 P2=3 Q1=2 Q2=2) (:AND P1=3 P2=3 Q1=1 Q2=1) (:AND P1=3 P2=3 Q1=0 Q2=0) (:AND P1=2 P2=2 Q1=2 Q2=2) (:AND P1=2 P2=2 Q1=1 Q2=1) (:AND P1=2 P2=2 Q1=0 Q2=0) (:AND P1=1 P2=1 Q1=2 Q2=2) (:AND P1=1 P2=1 Q1=1 Q2=1) (:AND P1=1 P2=1 Q1=0 Q2=0) (:AND P1=0 P2=0 Q1=2 Q2=2) (:AND P1=0 P2=0 Q1=1 Q2=1) (:AND P1=0 P2=0 Q1=0 Q2=0)) (:OR (:NOT PIPE2_IS_OK) (:AND P3=5 P4=5 Q3=2 Q4=2) (:AND P3=5 P4=5 Q3=1 Q4=1) (:AND P3=5 P4=5 Q3=0 Q4=0) (:AND P3=4 P4=4 Q3=2 Q4=2) (:AND P3=4 P4=4 Q3=1 Q4=1) (:AND P3=4 P4=4 Q3=0 Q4=0) (:AND P3=3 P4=3 Q3=2 Q4=2) (:AND P3=3 P4=3 Q3=1 Q4=1) (:AND P3=3 P4=3 Q3=0 Q4=0) (:AND P3=2 P4=2 Q3=2 Q4=2) (:AND P3=2 P4=2 Q3=1 Q4=1) (:AND P3=2 P4=2 Q3=0 Q4=0) (:AND P3=1 P4=1 Q3=2 Q4=2) (:AND P3=1 P4=1 Q3=1 Q4=1) (:AND P3=1 P4=1 Q3=0 Q4=0) (:AND P3=0 P4=0 Q3=2 Q4=2) (:AND P3=0 P4=0 Q3=1 Q4=1) (:AND P3=0 P4=0 Q3=0 Q4=0)) (:OR (:NOT PIPE3_IS_OK) (:AND P5=5 P6=5 Q5=2 Q6=2) (:AND P5=5 P6=5 Q5=1 Q6=1) (:AND P5=5 P6=5 Q5=0 Q6=0) (:AND P5=4 P6=4 Q5=2 Q6=2) (:AND P5=4 P6=4 Q5=1 Q6=1) (:AND P5=4 P6=4 Q5=0 Q6=0) (:AND P5=3 P6=3 Q5=2 Q6=2) (:AND P5=3 P6=3 Q5=1 Q6=1) (:AND P5=3 P6=3 Q5=0 Q6=0) (:AND P5=2 P6=2 Q5=2 Q6=2) (:AND P5=2 P6=2 Q5=1 Q6=1) (:AND P5=2 P6=2 Q5=0 Q6=0) (:AND P5=1 P6=1 Q5=2 Q6=2) (:AND P5=1 P6=1 Q5=1 Q6=1) (:AND P5=1 P6=1 Q5=0 Q6=0) (:AND P5=0 P6=0 Q5=2 Q6=2) (:AND P5=0 P6=0 Q5=1 Q6=1) (:AND P5=0 P6=0 Q5=0 Q6=0)) (:OR (:NOT PIPE4_IS_OK) (:AND P7=5 P8=5 Q7=2 Q8=2) (:AND P7=5 P8=5 Q7=1 Q8=1) (:AND P7=5 P8=5 Q7=0 Q8=0) (:AND P7=4 P8=4 Q7=2 Q8=2) (:AND P7=4 P8=4 Q7=1 Q8=1) (:AND P7=4 P8=4 Q7=0 Q8=0) (:AND P7=3 P8=3 Q7=2 Q8=2) (:AND P7=3 P8=3 Q7=1 Q8=1) (:AND P7=3 P8=3 Q7=0 Q8=0) (:AND P7=2 P8=2 Q7=2 Q8=2) (:AND P7=2 P8=2 Q7=1 Q8=1) (:AND P7=2 P8=2 Q7=0 Q8=0) (:AND P7=1 P8=1 Q7=2 Q8=2) (:AND P7=1 P8=1 Q7=1 Q8=1) (:AND P7=1 P8=1 Q7=0 Q8=0) (:AND P7=0 P8=0 Q7=2 Q8=2) (:AND P7=0 P8=0 Q7=1 Q8=1) (:AND P7=0 P8=0 Q7=0 Q8=0)) (:OR (:NOT CYL1_IS_OK) (:AND P2=5 P3=4 Q2=2 Q3=2 VCYL1=2) (:AND P2=5 P3=4 Q2=1 Q3=1 VCYL1=1) (:AND P2=4 P3=3 Q2=2 Q3=2 VCYL1=2) (:AND P2=4 P3=3 Q2=1 Q3=1 VCYL1=1) (:AND P2=3 P3=2 Q2=2 Q3=2 VCYL1=2) (:AND P2=3 P3=2 Q2=1 Q3=1 VCYL1=1) (:AND P2=2 P3=1 Q2=2 Q3=2 VCYL1=2) (:AND P2=2 P3=1 Q2=1 Q3=1 VCYL1=1) (:AND P2=1 P3=0 Q2=2 Q3=2 VCYL1=2) (:AND P2=1 P3=0 Q2=1 Q3=1 VCYL1=1) (:AND P2=5 P3=5 Q2=0 Q3=0 VCYL1=0) (:AND P2=4 P3=4 Q2=0 Q3=0 VCYL1=0) (:AND P2=3 P3=3 Q2=0 Q3=0 VCYL1=0) (:AND P2=2 P3=2 Q2=0 Q3=0 VCYL1=0) (:AND P2=1 P3=1 Q2=0 Q3=0 VCYL1=0) (:AND P2=0 P3=0 Q2=0 Q3=0 VCYL1=0)) (:OR (:NOT CYL2_IS_OK) (:AND P4=5 P5=4 Q4=2 Q5=2 VCYL2=2) (:AND P4=5 P5=4 Q4=1 Q5=1 VCYL2=1) (:AND P4=4 P5=3 Q4=2 Q5=2 VCYL2=2) (:AND P4=4 P5=3 Q4=1 Q5=1 VCYL2=1) (:AND P4=3 P5=2 Q4=2 Q5=2 VCYL2=2) (:AND P4=3 P5=2 Q4=1 Q5=1 VCYL2=1) (:AND P4=2 P5=1 Q4=2 Q5=2 VCYL2=2) (:AND P4=2 P5=1 Q4=1 Q5=1 VCYL2=1) (:AND P4=1 P5=0 Q4=2 Q5=2 VCYL2=2) (:AND P4=1 P5=0 Q4=1 Q5=1 VCYL2=1) (:AND P4=5 P5=5 Q4=0 Q5=0 VCYL2=0) (:AND P4=4 P5=4 Q4=0 Q5=0 VCYL2=0) (:AND P4=3 P5=3 Q4=0 Q5=0 VCYL2=0) (:AND P4=2 P5=2 Q4=0 Q5=0 VCYL2=0) (:AND P4=1 P5=1 Q4=0 Q5=0 VCYL2=0) (:AND P4=0 P5=0 Q4=0 Q5=0 VCYL2=0)) (:OR (:NOT VT_IS_OK) (:AND P6=5 P7=5 Q6=0 Q7=0) (:AND P6=5 P7=4 Q6=1 Q7=1) (:AND P6=5 P7=3 Q6=2 Q7=2) (:AND P6=4 P7=4 Q6=0 Q7=0) (:AND P6=4 P7=3 Q6=1 Q7=1) (:AND P6=4 P7=2 Q6=2 Q7=2) (:AND P6=3 P7=3 Q6=0 Q7=0) (:AND P6=3 P7=2 Q6=1 Q7=1) (:AND P6=3 P7=1 Q6=2 Q7=2) (:AND P6=2 P7=2 Q6=0 Q7=0) (:AND P6=2 P7=1 Q6=1 Q7=1) (:AND P6=2 P7=0 Q6=2 Q7=2) (:AND P6=1 P7=1 Q6=0 Q7=0) (:AND P6=1 P7=0 Q6=1 Q7=1) (:AND P6=0 P7=0 Q6=0 Q7=0)) (:OR (:NOT Q1=1) (:NOT Q1=2)) (:OR (:NOT Q1=0) (:NOT Q1=2)) (:OR (:NOT Q1=0) (:NOT Q1=1)) (:OR (:NOT P1=4) (:NOT P1=5)) (:OR (:NOT P1=3) (:NOT P1=5)) (:OR (:NOT P1=3) (:NOT P1=4)) (:OR (:NOT P1=2) (:NOT P1=5)) (:OR (:NOT P1=2) (:NOT P1=4)) (:OR (:NOT P1=2) (:NOT P1=3)) (:OR (:NOT P1=1) (:NOT P1=5)) (:OR (:NOT P1=1) (:NOT P1=4)) (:OR (:NOT P1=1) (:NOT P1=3)) (:OR (:NOT P1=1) (:NOT P1=2)) (:OR (:NOT P1=0) (:NOT P1=5)) (:OR (:NOT P1=0) (:NOT P1=4)) (:OR (:NOT P1=0) (:NOT P1=3)) (:OR (:NOT P1=0) (:NOT P1=2)) (:OR (:NOT P1=0) (:NOT P1=1)) (:OR (:NOT Q8=1) (:NOT Q8=2)) (:OR (:NOT Q8=0) (:NOT Q8=2)) (:OR (:NOT Q8=0) (:NOT Q8=1)) (:OR (:NOT P8=4) (:NOT P8=5)) (:OR (:NOT P8=3) (:NOT P8=5)) (:OR (:NOT P8=3) (:NOT P8=4)) (:OR (:NOT P8=2) (:NOT P8=5)) (:OR (:NOT P8=2) (:NOT P8=4)) (:OR (:NOT P8=2) (:NOT P8=3)) (:OR (:NOT P8=1) (:NOT P8=5)) (:OR (:NOT P8=1) (:NOT P8=4)) (:OR (:NOT P8=1) (:NOT P8=3)) (:OR (:NOT P8=1) (:NOT P8=2)) (:OR (:NOT P8=0) (:NOT P8=5)) (:OR (:NOT P8=0) (:NOT P8=4)) (:OR (:NOT P8=0) (:NOT P8=3)) (:OR (:NOT P8=0) (:NOT P8=2)) (:OR (:NOT P8=0) (:NOT P8=1)) (:OR (:NOT Q2=1) (:NOT Q2=2)) (:OR (:NOT Q2=0) (:NOT Q2=2)) (:OR (:NOT Q2=0) (:NOT Q2=1)) (:OR (:NOT P2=4) (:NOT P2=5)) (:OR (:NOT P2=3) (:NOT P2=5)) (:OR (:NOT P2=3) (:NOT P2=4)) ...

Erfüllbarkeitsalgorithmen

Q: Was hat die Beantwortung der Folgerungsfrage „Gilt $\alpha \models \beta$?“ mit dem Erfüllbarkeitsproblem zu tun?

A: Die Frage „Gilt $\alpha \models \beta$?“ lässt sich mit Hilfe eines Erfüllbarkeitsalgorithmus beantworten.

$$
\begin{aligned}
& \alpha \models \beta \\
\Leftrightarrow \quad & \forall \mathcal{I} : (\mathcal{I}(\alpha) = 1 \quad \Rightarrow \quad \mathcal{I}(\beta) = 1) \\
\Leftrightarrow \quad & \alpha \rightarrow \beta \text{ ist tautologisch} \\
\Leftrightarrow \quad & \alpha \wedge \neg\beta \text{ ist widerspruchsvoll / unerfüllbar}
\end{aligned}
$$

Bzw.:

$$
\begin{aligned}
& \alpha \not\models \beta \\
\Leftrightarrow \quad & \text{Nicht } (\alpha \wedge \neg\beta \text{ unerfüllbar}) \\
\Leftrightarrow \quad & \alpha \wedge \neg\beta \text{ erfüllbar}
\end{aligned}
$$

Erfüllbarkeitsalgorithmen

Verschiedene Möglichkeiten, die Erfüllbarkeit einer Formel α zu entscheiden.

Erstellen einer Wahrheitstafel

Vorteil: beliebige Formelstruktur, beliebige Junktoren

Nachteil: auch in einfachen Fällen exponentiell,
keine Berücksichtigung der Formelstruktur

Erfüllbarkeitsalgorithmen

Verschiedene Möglichkeiten, die Erfüllbarkeit einer Formel α zu entscheiden.

Erstellen einer Wahrheitstafel

Vorteil: beliebige Formelstruktur, beliebige Junktoren

Nachteil: auch in einfachen Fällen exponentiell,
keine Berücksichtigung der Formelstruktur

Analyse der Formelstruktur

(a) Top-down-Auswertung des Formelbaumes mit Fallunterscheidung bei Alternativen.

(b) systematische Anwendung mit Darstellung als Baum: analytisches Tableau

Vorteil: beliebige Formelstruktur, beliebige Junktoren

Nachteil: auch in einfachen Fällen exponentiell,
umfangreiche Implementation

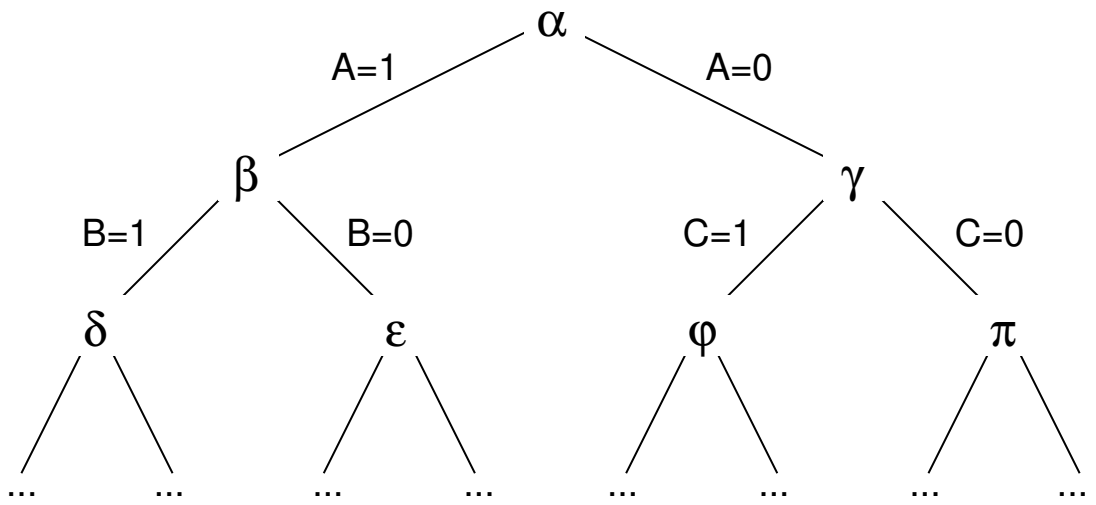
Erfüllbarkeitsalgorithmen

Semantische Bäume

Kombination aus der Überprüfung von Bewertungen und Analyse der Formelstruktur.

Vorteil: sukzessive Auswertung orientiert sich an Formelstruktur

Nachteil: reduzierte (=teil-ausgewertete) Formel relativ kompliziert zu berechnen



Bemerkungen:

- Die Bewertung eines Atoms A mit 0 bzw. 1 kann auf Formelebene dadurch nachgebildet werden, dass A durch die widerspruchsvolle Formel $T \wedge \neg T$ bzw. die tautologische Formel $T \vee \neg T$ substituiert wird. (T beliebig, aber fest, für alle Ersetzungen gleich wählbar.)
- Die reduzierte Formel ist die Formel, die sich aufgrund der möglichen Vereinfachungen aus einer entsprechend substituierten Formel ergibt. Die Vereinfachungsregeln folgen auf der nächsten Folie.

Semantische Bäume

Definition 20 (1-Äquivalenzen, 0-Äquivalenzen)

1. $\neg(\beta \vee \neg\beta) \approx (\beta \wedge \neg\beta)$,
   $\neg(\beta \wedge \neg\beta) \approx (\beta \vee \neg\beta)$

2. $\alpha \vee (\beta \vee \neg\beta) \approx (\beta \vee \neg\beta) \vee \alpha \approx (\beta \vee \neg\beta)$,
   $\alpha \vee (\beta \wedge \neg\beta) \approx (\beta \wedge \neg\beta) \vee \alpha \approx \alpha$

3. $\alpha \wedge (\beta \vee \neg\beta) \approx (\beta \vee \neg\beta) \wedge \alpha \approx \alpha$,
   $\alpha \wedge (\beta \wedge \neg\beta) \approx (\beta \wedge \neg\beta) \wedge \alpha \approx (\beta \wedge \neg\beta)$

4. $\alpha \rightarrow (\beta \vee \neg\beta) \approx (\beta \vee \neg\beta)$,
   $\alpha \rightarrow (\beta \wedge \neg\beta) \approx \neg\alpha$,
   $(\beta \vee \neg\beta) \rightarrow \alpha \approx \alpha$,
   $(\beta \wedge \neg\beta) \rightarrow \alpha \approx (\beta \vee \neg\beta)$

5. $\alpha \leftrightarrow (\beta \wedge \neg\beta) \approx (\beta \wedge \neg\beta) \leftrightarrow \alpha \approx \neg\alpha$,
   $\alpha \leftrightarrow (\beta \vee \neg\beta) \approx (\beta \vee \neg\beta) \leftrightarrow \alpha \approx \alpha$

Semantische Bäume

Definition 21 (1-Reduktion, 0-Reduktion)

Sei α eine aussagenlogische Formel und $A \in \mathit{atoms}(\alpha)$. Weiterhin sei β das Ergebnis der Ersetzung jedes Vorkommens von A in α durch $A \vee \neg A$. Dann bezeichne

$$\alpha[A/1]$$

eine kürzeste Formel, die aus β unter Anwendung der 1- bzw. 0-Äquivalenzen entstehen kann. (1-Reduktion)

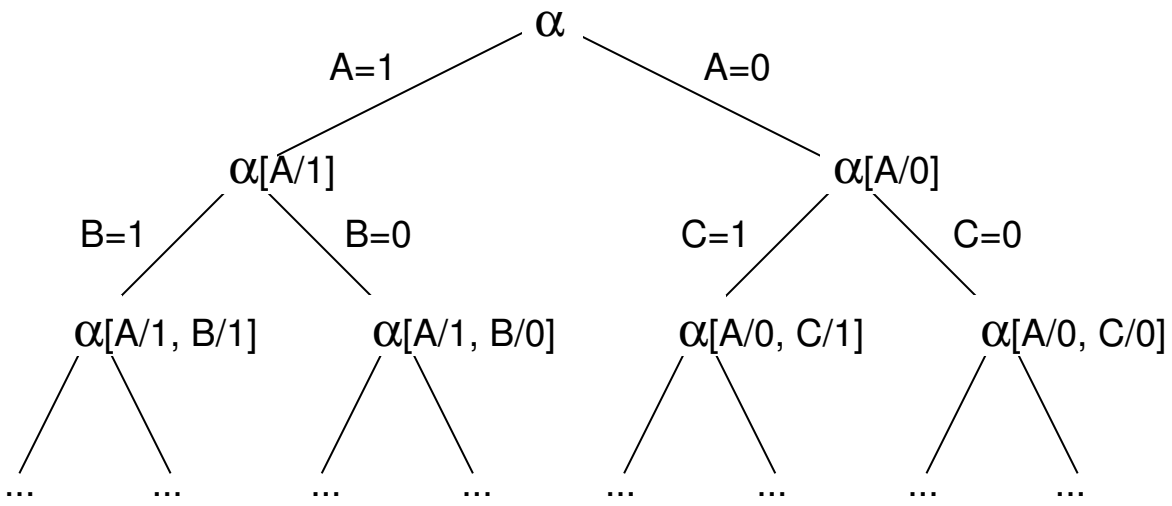
Sei α eine aussagenlogische Formel und $A \in \mathit{atoms}(\alpha)$. Weiterhin sei β das Ergebnis der Ersetzung jedes Vorkommens von A in α durch $A \wedge \neg A$. Dann bezeichne

$$\alpha[A/0]$$

eine kürzeste Formel, die aus β unter Anwendung der 1- bzw. 0-Äquivalenzen entstehen kann. (0-Reduktion)

Mehrfache Reduktionen hintereinander werden in einer Klammer zusammengefasst. Beispiel: $\alpha[A/1, B/1, C/0] := ((\alpha[A/1])[B/1])[C/0]$

Semantische Bäume



Lemma 12 (Splitting Regel)

Für eine aussagenlogische Formel α und ein Atom $A \in \mathit{atoms}(\alpha)$ gilt

α erfüllbar $\Leftrightarrow$ $\alpha[A/1]$ erfüllbar
oder
$\alpha[A/0]$ erfüllbar

Semantische Bäume

Beschränkung der Formelstruktur: Sei $\alpha \in$ KNF, $A \in \mathit{atoms}(\alpha)$

Algorithmus: SPLIT-SAT
Input: α. A formula in CNF.
Output: sat. A flag indicating whether α is satisfiable.

```
SPLIT-SAT(α)
  IF α = β ∧ ¬β AND β is prime formula
  THEN RETURN('FALSE')
  IF α = β ∨ ¬β AND β is prime formula
  THEN RETURN('TRUE')
  A = choose(atoms(α))
  IF SPLIT-SAT(α[A/1])
  THEN RETURN('TRUE')
  ELSE RETURN SPLIT-SAT(α[A/0])
```

Fragen:

- Welche Suchstrategie verfolgt SPLIT-SAT?
- Geht SPLIT-SAT systematisch vor?

Weiterentwicklung semantischer Bäume

Verbesserung von `SPLIT-SAT`.

- Idee 1:

  Tritt in einer Formel $\alpha \in$ KNF eine Unitklausel L auf, so muss L mit 1 bewertet werden.

  Stichwort: Unit-Reduktion

- Idee 2:

  Tritt in einer Formel $\alpha \in$ KNF ein Atom A nur in positiven oder nur in negativen Literalen auf, so kann das Literal mit 1 bewertet werden, um α zu erfüllen.

  Stichwort: Pure-Literal-Reduktion

Bemerkung: Beachte den Unterschied zwischen „muss“ und „kann“.

Weiterentwicklung semantischer Bäume

Verallgemeinerung von $\alpha[A/1]$.

Sei L ein Literal über *atoms*(α).

- Dann gilt für $L = A$: $\alpha[L/1] := \alpha[A/1]$
- Dann gilt für $L = \neg A$: $\alpha[L/1] = \alpha[\neg A/1] := \alpha[A/0]$

Mit $\neg L$ meinen wir das komplementäre Literal, d.h. für $L = \neg A$ sei $\neg L = A$.

Beispiel:
Sei $\alpha = \neg A$ und $L = \neg A$.
$\alpha[L/1] = \alpha[\neg A/1] = \alpha[A/0] \approx \neg(T \wedge \neg T) \approx T \vee \neg T$

Man ersetzt quasi Literal und Komplement durch die Formeln für die Wahrheitswerte.

Algorithmische Hinweise zur Realisierung von `SPLIT-SAT`

- Datenstruktur: Listen von Listen von Literalen
- Berechnung von $\alpha[L/1]$:
  1. Streiche Klauseln mit Literal L.
  2. Streiche $\neg L$ in den verbliebenen Klauseln.
  3. α erfüllbar, falls Klauselliste leer; falls eine Literalliste leer, Backtracking.

Weiterentwicklung semantischer Bäume

Davis-Putnam-Algorithmus [Davis/Putnam 1960, Davis/Loveland/Logemann 1962]

1. Unit-Reduktion
2. Pure-Literal-Reduktion
3. Splitting

Algorithmus: DPLL-SAT
Input: α. A formula in CNF.
Output: sat. A flag indicating whether α is satisfiable.

```
DPLL-SAT(α)
  IF α = β ∧ ¬β AND β is prime formula
  THEN RETURN('FALSE')
  IF α = β ∨ ¬β AND β is prime formula
  THEN RETURN('TRUE')

  IF units(α) ≠ ∅
  THEN L = choose(units(α)), RETURN(DPLL-SAT(α[L/1]))
  IF pures(α) ≠ ∅
  THEN L = choose(pures(α)), RETURN(DPLL-SAT(α[L/1]))

  A = choose(atoms(α))
  IF DPLL-SAT(α[A/1])
  THEN RETURN('TRUE')
  ELSE RETURN DPLL-SAT(α[A/0])
```

Weiterentwicklung semantischer Bäume

Beispiel für `DPLL-SAT`:

$\alpha = (A \vee \neg B \vee \neg C) \wedge (\neg A \vee B \vee C) \wedge$
$(\neg B \vee C) \wedge (B \vee \neg C) \wedge (\neg B \vee \neg C) \wedge (\neg F \vee C)$

$\alpha[\neg F/1] = (A \vee \neg B \vee \neg C) \wedge (\neg A \vee B \vee C) \wedge (\neg B \vee C) \wedge (B \vee \neg C) \wedge (\neg B \vee \neg C)$

$\alpha[\neg F/1, A/1] = (B \vee C) \wedge (\neg B \vee C) \wedge (B \vee \neg C) \wedge (\neg B \vee \neg C)$

$\alpha[\neg F/1, A/0] = (\neg B \vee \neg C) \wedge (\neg B \vee C) \wedge (B \vee \neg C) \wedge (\neg B \vee \neg C)$

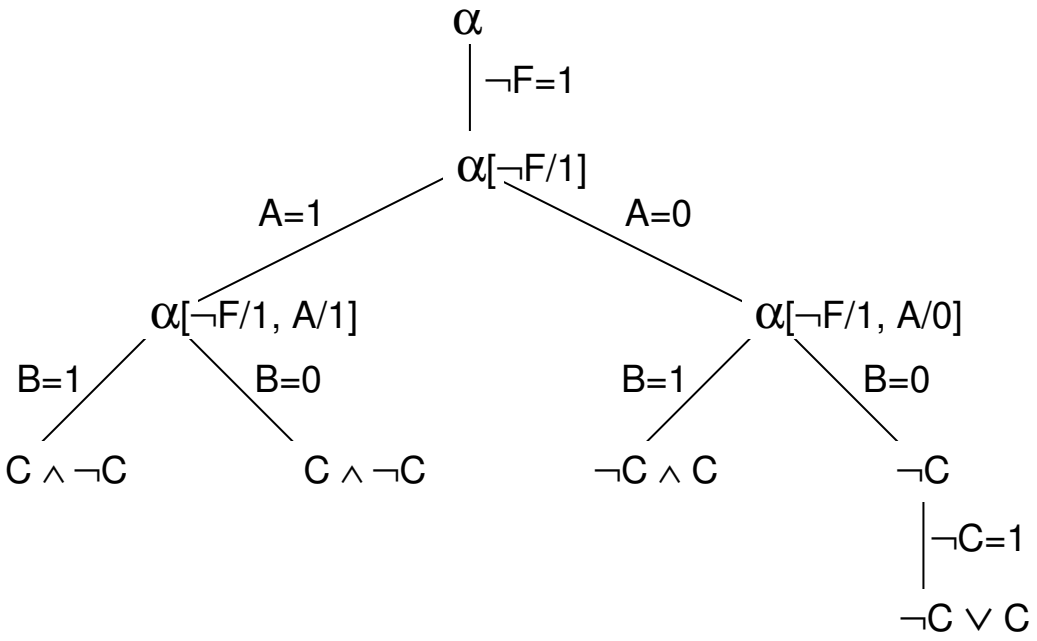
$\alpha[\neg F/1, A/1, B/1] = C \wedge \neg C$

$\alpha[\neg F/1, A/1, B/0] = C \wedge \neg C$

$\alpha[\neg F/1, A/0, B/1] = \neg C \wedge C$

$\alpha[\neg F/1, A/0, B/0] = \neg C$

$\alpha[\neg F/1, A/0, B/0, C/1] = \neg C \vee C$



Weiterentwicklung semantischer Bäume

Auswahlkriterien für Splittingregel:

- Erstes Vorkommen des Atoms bzw. Literals (systematische Suche)
- häufigstes Atom bzw. Literal
  (gute Heuristik, falls Wahrscheinlichkeit für Erfüllbarkeit der Formel hoch)
- Atom mit größter Differenz aus Anzahl positiver und negativer Vorkommen

Weiterentwicklung semantischer Bäume

Auswahlkriterien für Splittingregel:

- Erstes Vorkommen des Atoms bzw. Literals (systematische Suche)
- häufigstes Atom bzw. Literal
  (gute Heuristik, falls Wahrscheinlichkeit für Erfüllbarkeit der Formel hoch)
- Atom mit größter Differenz aus Anzahl positiver und negativer Vorkommen

- van Geldern:
  a) Mehrere Atome mit mehr als 3 Vorkommen vorhanden:
     Wähle Atom mit maximalem Produkt aus Anzahl positiver und negativer Vorkommen.
  b) Wähle Atom mit maximalem Vorkommen (2 oder 3).

Weiterentwicklung semantischer Bäume

Auswahlkriterien für Splittingregel:

- Erstes Vorkommen des Atoms bzw. Literals (systematische Suche)
- häufigstes Atom bzw. Literal
  (gute Heuristik, falls Wahrscheinlichkeit für Erfüllbarkeit der Formel hoch)
- Atom mit größter Differenz aus Anzahl positiver und negativer Vorkommen

- van Geldern:
  a) Mehrere Atome mit mehr als 3 Vorkommen vorhanden:
     Wähle Atom mit maximalem Produkt aus Anzahl positiver und negativer Vorkommen.
  b) Wähle Atom mit maximalem Vorkommen (2 oder 3).

- SAT-Winner (Paderborn, 1991):
  $h_i(L)$ = Anzahl Klauseln der Länge i mit Literal L.
  $H_i(A) = \max(h_i(A), h_i(\neg A)) + 2 \cdot \min(h_i(A), h_i(\neg A))$
  Wähle Atom A mit lexikographisch größtem Vektor

$$(H_2(A), H_3(A), \ldots, H_n(A))$$

Kapitel L: II (Fortsetzung)

II. Aussagenlogik

- Syntax der Aussagenlogik
- Semantik der Aussagenlogik
- Eigenschaften des Folgerungsbegriffs
- Äquivalenz
- Formeltransformation
- Normalformen

- Bedeutung der Folgerung
- Erfüllbarkeitsalgorithmen
- Semantische Bäume
- Weiterentwicklung semantischer Bäume
- Syntaktische Schlussfolgerungsverfahren
- Erfüllbarkeitsprobleme

Syntaktische Schlussfolgerungsverfahren

Ausgangspunkt war die Frage: „Gilt $\alpha \models \beta$?“

Bisher wurde die Antwort auf diese Frage durch Anwendung der Folgerungsdefinition gefunden – d. h., durch Auswerten der möglichen Interpretationen.

Q: Lässt sich obige Frage auch *ohne* Rückgriff auf die Semantik, d. h. ohne Auswertung der Interpretationen beantworten?

A: Ja. Wurde für eine bestimmte Formelstruktur α festgestellt, dass $\alpha \models \beta$ gilt, so reicht in Zukunft das Identifizieren dieser Struktur aus, um die Folgerbarkeit von β zu behaupten.

Diese Überlegung führt zum Begriff der Herleitung.

Syntaktische Schlussfolgerungsverfahren

Definition 22 (Schlussregel, Kalkül)

Unter einer Schlussregel versteht man ein Verfahren (einen Mechanismus, eine Vorschrift) zur Erzeugung einer neuen Formel aus vorhandenen Formeln, das sich nur an syntaktischen Eigenschaften der beteiligten Formeln orientiert. Stichwort: „Zeichenkettenmanipulationen“

Ein Kalkül bezeichnet eine Menge von Schlussregeln.

Idee der Herleitung

Sei $\alpha = \alpha_1 \wedge \alpha_2 \wedge \ldots \wedge \alpha_n$.

$\beta_1, \beta_2, \ldots, \beta_n$ ist eine Herleitung von β aus α genau dann, wenn gilt:

1. $\beta_n = \beta$ und
2. für jedes $i \in \{1, 2, \ldots, n\}$ gilt
   (a) $\beta_i \in \{\alpha_1, \ldots, \alpha_n\}$ oder
   (b) β_i wird mit einer Schlussregel hergeleitet, deren Voraussetzungen aus $\beta_1, \beta_2, \ldots, \beta_{i-1}$ stammen.

Bemerkungen:

- Im allgemeinen beschreibt eine Schlussregel ein Schema, das Variablen für Formelnverwendet anstelle von konkreten (aussagenlogischen) Formeln. Eine Schlussregel ist damit genauso zu handhaben wie die Äquivalenzen auf S. II- 47 und S. II- 96 . Wenn konkrete Formeln die in der Regel verlangte Struktur aufweisen, kann die Regel angewendet werden. Die Schlussregel fasst damit die abzählbar vielen denkbaren Einzelsituationen mit konkreten Formeln zu einem Schema zusammen.

Syntaktische Schlussfolgerungsverfahren

Definition 23 (Herleiten, syntaktisches Schlussfolgern)

Sei α eine Formel, die uns z. B. als Modell eines Weltausschnittes dient. Sei β eine weitere Formel.

Die Erzeugung von β auf Basis eines Kalküls wird als Herleiten oder als syntaktisches Schlussfolgern von β bezeichnet.

In Zeichen: $\alpha \vdash \beta$

Bemerkungen:

- Man ist an Kalkülen interessiert, mit denen man Folgerungen herleiten kann. (klar)
- Wiederholung: Im Gegensatz zum *Herleiten* von β wird die Erzeugung von β auf Basis einer semantischen Analyse als *Folgern* oder als *semantisches Schlussfolgern* bezeichnet. In Zeichen: $\alpha \models \beta$

Frage:

Ist `DPLL-SAT` ein syntaktisches oder ein semantische Verfahren zum Schlussfolgern?

Syntaktische Schlussfolgerungsverfahren

Die Schlussregel *„Modus Ponens“*, MP (drei verschiedene Schreibweisen)

$$\frac{\gamma \quad \gamma \rightarrow \beta}{\beta} \qquad \frac{\gamma \wedge (\gamma \rightarrow \beta)}{\beta} \qquad \frac{\gamma,\ \gamma \rightarrow \beta}{\beta}$$

γ und $\gamma \rightarrow \beta$ sind die Voraussetzungen des MP, β die Konklusion.

Es handelt um eine rein syntaktische Vorschrift: Der MP kann angewandt werden, wenn genau die beschriebene syntaktische Struktur vorgefunden wird.

Syntaktische Schlussfolgerungsverfahren

Die Schlussregel „*Modus Ponens*“, MP (drei verschiedene Schreibweisen)

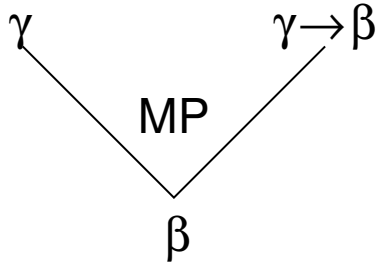
$$\frac{\gamma \quad \gamma \rightarrow \beta}{\beta} \qquad \frac{\gamma \wedge (\gamma \rightarrow \beta)}{\beta} \qquad \frac{\gamma, \gamma \rightarrow \beta}{\beta}$$

γ und $\gamma \rightarrow \beta$ sind die Voraussetzungen des MP, β die Konklusion.

Es handelt um eine rein syntaktische Vorschrift: Der MP kann angewandt werden, wenn genau die beschriebene syntaktische Struktur vorgefunden wird.

Sei $\alpha = \gamma \wedge (\gamma \rightarrow \beta)$. Anwendung der Schlussregel MP:

- In Zeichen: $\alpha \mid\!\!\frac{}{MP} \beta$
- In Worten: „β kann mittels MP aus α hergeleitet werden.“
- Als Graphik:



Bemerkungen:

- Falls es sich bei der hergeleiteten Formel β um eine Folgerung aus α handelt, kann β konjunktiv zu α hinzugenommen werden, wobei die logische Äquivalenz zu α erhalten bleibt: $\alpha \approx \alpha \wedge \beta$

Syntaktische Schlussfolgerungsverfahren

Zunächst stellt sich allgemein die Frage der Korrektheit.

- Handelt es sich bei allen Formeln, die mittels einer Schlussregel herleitbar sind, um Folgerungen?

  In Zeichen: Gilt $\alpha \vdash \beta \quad \Rightarrow \quad \alpha \models \beta$?

Frage: Wie ist die Korrektheit einer Schlussregel nachweisbar?

Syntaktische Schlussfolgerungsverfahren

Zunächst stellt sich allgemein die Frage der Korrektheit.

- Handelt es sich bei allen Formeln, die mittels einer Schlussregel herleitbar sind, um Folgerungen?

  In Zeichen: Gilt $\alpha \vdash \beta \quad \Rightarrow \quad \alpha \models \beta$?

Frage: Wie ist die Korrektheit einer Schlussregel nachweisbar?

Weiterhin stellt sich die Frage nach der Güte eines Kalküls, der sogenannten Vollständigkeit.

- Wie viele oder welche der Folgerungen sind mittels eines Kalküls herleitbar?

  In Zeichen: Gilt $\alpha \models \beta \quad \Rightarrow \quad \alpha \vdash \beta$?

Frage: Wie ist die Vollständigkeit eines Kalküls nachweisbar?

Syntaktische Schlussfolgerungsverfahren

Definition 24 (Korrektheit, Vollständigkeit eines Kalküls)

Sei α eine Formel. Ein Kalkül (ein Verfahren zur Erzeugung von Formeln) heißt *korrekt*, wenn jede Formel β, die auf Basis des Kalküls aus α erzeugt wurde, eine Folgerung aus α ist.

Ein Kalkül heißt *vollständig*, wenn jede Formel β, die sich aus α folgern lässt, hergeleitet werden kann.

Bemerkungen:

- Mit der Korrektheit und Vollständigkeit zeigt man für einen Kalkül die *Äquivalenz von Syntax und Semantik*:
  1. Syntax $\Rightarrow$ Semantik: Jede Herleitung ist eine Folgerung.
  2. Semantik $\Rightarrow$ Syntax: Jede Folgerung lässt sich herleiten.
- Die Korrektheit kann für jede Schlussregel isoliert verifiziert werden.

Syntaktische Schlussfolgerungsverfahren

Korrektheit der Schlussregel MP

(a) Zu $\alpha \models \beta$ äquivalent ist:
- $\alpha \approx \{\alpha, \beta\}$ bzw. $\alpha \wedge \beta$
- $\alpha \rightarrow \beta$ ist tautologisch
- $\{\alpha, \neg\beta\}$ bzw. $\alpha \wedge \neg\beta$ ist widerspruchsvoll
- ...

(b) Sei $\alpha = \{\gamma, \gamma \rightarrow \beta\}$. Anwendung des MP: $\alpha \mid\!\!\overline{}_{MP} \beta$

(c) Zu zeigen (eines reicht):
- $\{\gamma, \gamma \rightarrow \beta\} \approx \{\gamma, \gamma \rightarrow \beta, \beta\}$
- $(\gamma \wedge (\gamma \rightarrow \beta)) \rightarrow \beta$ ist tautologisch
- $\{\gamma, \gamma \rightarrow \beta, \neg\beta\}$ ist widerspruchsvoll
- ...

(d) Überprüfung aller Interpretationen mit Hilfe der Wahrheitstafel.
Ergebnis: β ist eine Folgerung aus α. Somit ist der MP eine korrekte Schlussregel.

Syntaktische Schlussfolgerungsverfahren

Vollständigkeit des MP

Unter alleiniger Verwendung des MP können nicht alle Folgerungen aus einer Formel α hergeleitet werden.

Syntaktische Schlussfolgerungsverfahren

Vollständigkeit des MP

Unter alleiniger Verwendung des MP können nicht alle Folgerungen aus einer Formel α hergeleitet werden.

Beispiel:

$\alpha = \{\gamma, \neg\gamma \vee \beta\}$. Es gilt: $\alpha \models \beta$

Die Formel α bietet keine Möglichkeit, den MP anzuwenden.

Ausweg:

Hinzunahme voraussetzungsloser Schlussregeln mit denen sich nur Tautologien herleiten lassen. Beispiel (Axiom A_4):

$$\frac{\{\}}{(\neg\varphi \vee \psi) \;\rightarrow\; (\varphi \rightarrow \psi)}$$

Bemerkungen:

- Für eine Tautologie δ gilt: $\alpha \approx \{\alpha, \delta\}$ bzw. $\alpha \approx \alpha \wedge \delta$
  (vgl. auch Definition 20, S. 96)
- Ziel ist es, den MP auf $\{\alpha, \delta\}$ anwenden zu können.
- Voraussetzungslose Schlussregeln werden Axiome genannt.
- Durch Hinzunahme der entsprechenden Axiome (Frege-Axiome) wird die Schlussregel MP zu einem vollständigen Kalkül, dem sogenannten Hilbert-Kalkül.

  Eine mögliche Version der Frege-Axiome des Hilbert-Kalküls:

A_1:	$(\varphi \leftrightarrow \psi) \rightarrow ((\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi))$	Definition $\leftrightarrow$
A_2:	$((\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)) \rightarrow (\varphi \leftrightarrow \psi)$	Definition $\leftrightarrow$
A_3:	$(\varphi \rightarrow \psi) \rightarrow (\neg\varphi \vee \psi)$	Definition $\rightarrow$
A_4:	$(\neg\varphi \vee \psi) \rightarrow (\varphi \rightarrow \psi)$	Definition $\rightarrow$
A_5:	$(\varphi \vee \psi) \rightarrow \neg(\neg\varphi \wedge \neg\psi)$	Definition $\wedge$
A_6:	$(\varphi \wedge \psi) \rightarrow \neg(\neg\varphi \vee \neg\psi)$	Definition $\wedge$
A_7:	$(\varphi \vee \varphi) \rightarrow \varphi$	Idempotenz von $\vee$
A_8:	$\varphi \rightarrow (\varphi \vee \psi)$	Abschwächung
A_9:	$(\varphi \vee \psi) \rightarrow (\psi \vee \varphi)$	Kommutativität von $\vee$
A_{10}:	$(\varphi \rightarrow \psi) \rightarrow ((\chi \rightarrow \varphi) \rightarrow (\chi \rightarrow \psi))$	

Syntaktische Schlussfolgerungsverfahren

Beispiel: Herleitung von β aus $\alpha = \gamma \wedge (\neg\gamma \vee \beta)$.

Syntaktische Schlussfolgerungsverfahren

Beispiel: Herleitung von β aus $\alpha = \gamma \wedge (\neg\gamma \vee \beta)$.

(a) Unter Anwendung des Axioms A_4 entsteht die Herleitung:
$(1)\ \gamma, \quad (2)\ \neg\gamma \vee \beta, \quad (3)\ (\neg\gamma \vee \beta) \rightarrow (\gamma \rightarrow \beta)$

Syntaktische Schlussfolgerungsverfahren

Beispiel: Herleitung von β aus $\alpha = \gamma \wedge (\neg\gamma \vee \beta)$.

(a) Unter Anwendung des Axioms A_4 entsteht die Herleitung:
$(1)\ \gamma, \quad (2)\ \neg\gamma \vee \beta, \quad (3)\ (\neg\gamma \vee \beta) \rightarrow (\gamma \rightarrow \beta)$

(b) Erste Anwendung des MP:

$$\frac{\begin{array}{r}\neg\gamma \vee \beta \\ (\neg\gamma \vee \beta) \rightarrow (\gamma \rightarrow \beta)\end{array}}{\gamma \rightarrow \beta}$$

. . . führt zur Herleitung:
$(1)\ \gamma, \quad (2)\ \neg\gamma \vee \beta, \quad (3)\ (\neg\gamma \vee \beta) \rightarrow (\gamma \rightarrow \beta), \quad (4)\ \gamma \rightarrow \beta$

Syntaktische Schlussfolgerungsverfahren

Beispiel: Herleitung von β aus $\alpha = \gamma \wedge (\neg\gamma \vee \beta)$.

(a) Unter Anwendung des Axioms A_4 entsteht die Herleitung:
$(1)\ \gamma, \quad (2)\ \neg\gamma \vee \beta, \quad (3)\ (\neg\gamma \vee \beta) \rightarrow (\gamma \rightarrow \beta)$

(b) Erste Anwendung des MP:

$$\frac{\begin{array}{r}\neg\gamma \vee \beta \\ (\neg\gamma \vee \beta) \rightarrow (\gamma \rightarrow \beta)\end{array}}{\gamma \rightarrow \beta}$$

...führt zur Herleitung:
$(1)\ \gamma, \quad (2)\ \neg\gamma \vee \beta, \quad (3)\ (\neg\gamma \vee \beta) \rightarrow (\gamma \rightarrow \beta), \quad (4)\ \gamma \rightarrow \beta$

(c) Zweite Anwendung des MP:

$$\frac{\begin{array}{r}\gamma \\ \gamma \rightarrow \beta\end{array}}{\beta}$$

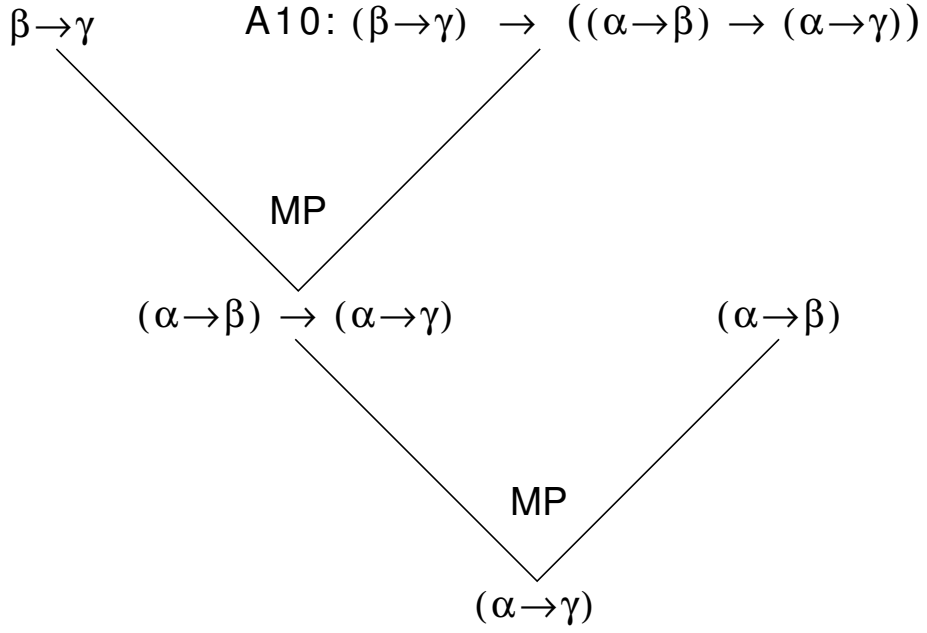
...führt zur gewünschten Herleitung:
$(1)\ \gamma, \quad (2)\ \neg\gamma \vee \beta, \quad (3)\ (\neg\gamma \vee \beta) \rightarrow (\gamma \rightarrow \beta), \quad (4)\ \gamma \rightarrow \beta, \quad (5)\ \beta$

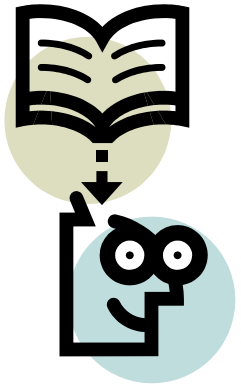
Syntaktische Schlussfolgerungsverfahren

Weitere Schlussregeln sind denkbar – z. B. der Kettenschluss, KS:

$$\frac{\alpha \rightarrow \beta,\ \ \beta \rightarrow \gamma}{\alpha \rightarrow \gamma}$$

Der KS ist praktisch, doch er ist keine echte Erweiterung des MP, sondern nur dessen zweifache Anwendung einschließlich der Verwendung von Axiom A_{10}:





Syntaktische Schlussfolgerungsverfahren

Die Schlussregel „Resolution“ (Formeln in KNF, Mengenschreibweise)

Seien α und β Klauseln mit $L \in \alpha$ und $\neg L \in \beta$.

$$\frac{\alpha \qquad\qquad \beta}{(\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\})}$$

α und β heißen Elternklauseln,

$(\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\})$ heißt Resolvente.

Syntaktische Schlussfolgerungsverfahren

Die Schlussregel „Resolution“ (Formeln in KNF, Mengenschreibweise)

Seien α und β Klauseln mit $L \in \alpha$ und $\neg L \in \beta$.

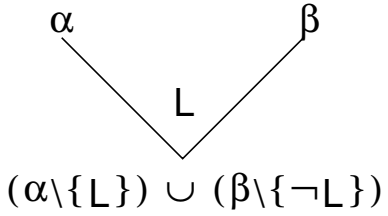
$$\frac{\alpha \qquad\qquad \beta}{(\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\})}$$

α und β heißen Elternklauseln,

$(\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\})$ heißt Resolvente.

Anwendung der Schlussregel Resolution:

- In Zeichen: $\alpha, \beta \mid\!\!\frac{1}{Res} (\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\})$
- In Worten: „$(\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\})$ kann in einem Schritt mittels der Resolutionsregel aus α und β hergeleitet werden.“
- Als Graphik:



Bemerkungen:

- Falls die Resolvente eine Folgerung der Elternklauseln ist, kann sie konjunktiv zur ihnen hinzugenommen werden, wobei die logische Äquivalenz erhalten bleibt:
  $\alpha \wedge \beta \;\approx\; \alpha \wedge \beta \wedge ((\alpha \setminus \{L\}) \cup (\beta \setminus \{\neg L\}))$

Syntaktische Schlussfolgerungsverfahren

Beispiele:

$\{A\},\ \{\neg A, B\} \mid\!\!\frac{1}{Res}\ \{B\}$

$\{\neg A, B\},\ \{\neg B\} \mid\!\!\frac{1}{Res}\ \{\neg A\}$

$\{A\},\ \{\neg A\} \mid\!\!\frac{1}{Res}\ \{\}$ (leere Klausel)

Vereinbarung:

Die leere Klausel wird als kleinste widersprüchliche Klausel aufgefasst – in Zeichen: $A \wedge \neg A$ bzw. $\{\}$ bzw. $\sqcup$.

Die leere Klausel ist das Kennzeichen für das Vorhandensein zweier komplementärer Unit-Klauseln. Sie kann Bestandteil einer Formel oder durch eine Herleitung erzeugt sein.

Syntaktische Schlussfolgerungsverfahren

Definition 25 (Resolutionskalkül)

Sei $\alpha = \{\alpha_1, \ldots, \alpha_n\}$ eine Formel in KNF und sei π eine Klausel.

π is mittels der Resolution aus α herleitbar, in Zeichen: $\alpha \mid\!\!\frac{}{Res} \pi$, genau dann, wenn es eine Folge von Klauseln $\pi_1, \ldots, \pi_k$ gibt mit

- $\pi_k = \pi$ und
- für alle $j = 1, \ldots, k$ gibt es Klauseln $\sigma_j, \tau_j \in \alpha \cup \{\pi_1, \ldots, \pi_k\}$ mit $\sigma_j, \tau_j \mid\!\!\frac{1}{Res} \pi_j$ oder $\pi_j \in \alpha$.

$\pi_1, \ldots, \pi_k$ heißt Herleitung von π aus α. Die Länge der Herleitung ist k.

Syntaktische Schlussfolgerungsverfahren

Definition 25 (Resolutionskalkül)

Sei $\alpha = \{\alpha_1, \ldots, \alpha_n\}$ eine Formel in KNF und sei π eine Klausel.

π is mittels der Resolution aus α herleitbar, in Zeichen: $\alpha \mid\!\!\frac{}{Res} \pi$, genau dann, wenn es eine Folge von Klauseln $\pi_1, \ldots, \pi_k$ gibt mit

- $\pi_k = \pi$ und
- für alle $j = 1, \ldots, k$ gibt es Klauseln $\sigma_j, \tau_j \in \alpha \cup \{\pi_1, \ldots, \pi_k\}$ mit $\sigma_j, \tau_j \mid\!\!\frac{1}{Res} \pi_j$ oder $\pi_j \in \alpha$.

$\pi_1, \ldots, \pi_k$ heißt Herleitung von π aus α. Die Länge der Herleitung ist k.

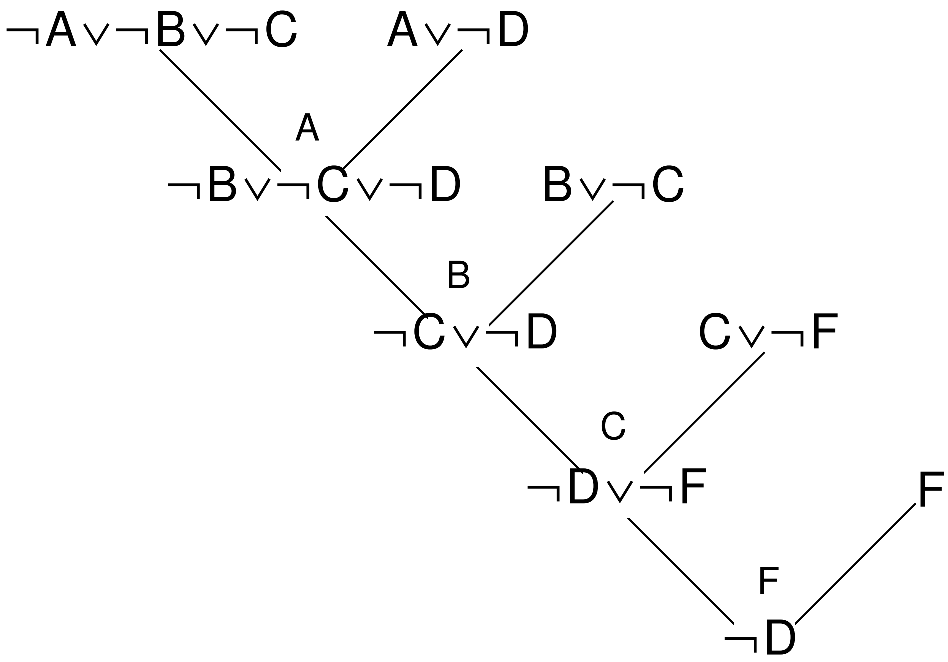
Schreibweisen:

- $\alpha \mid\!\!\frac{}{Res} \pi$
- $\alpha_1, \ldots, \alpha_n \mid\!\!\frac{}{Res} \pi$
- $\alpha_1, \ldots, \alpha_n \mid\!\!\frac{k}{Res} \pi$
- auch: $\alpha \mid\!\!\frac{}{Res} \beta$ mit $\alpha, \beta \in$ KNF

Syntaktische Schlussfolgerungsverfahren

Beispiel (Input-Resolution).

$\alpha = \{\neg A \vee \neg B \vee \neg C, A \vee \neg D, B \vee \neg C, C \vee \neg F, F\}$



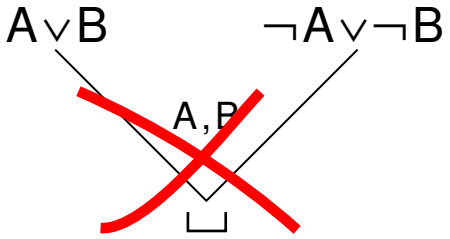
Offensichtlich gilt: $\alpha \mid\!\frac{}{Res} \neg D$

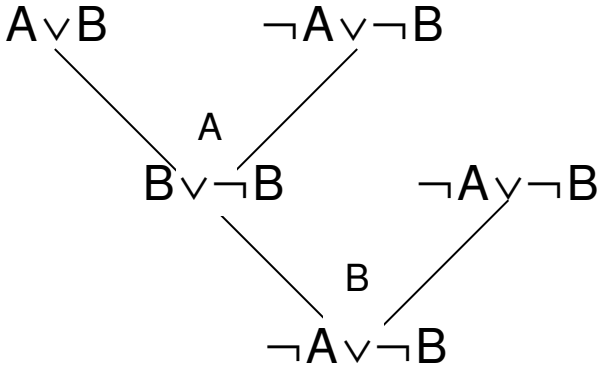
Begriff: Herleitungsbaum oder auch Beweisbaum (ist immer ein DAG)

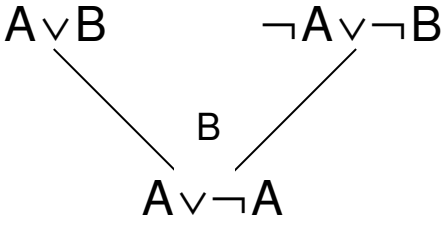
Syntaktische Schlussfolgerungsverfahren

Beispiel.

$\alpha = \{A \vee B, \neg A \vee \neg B\}$.





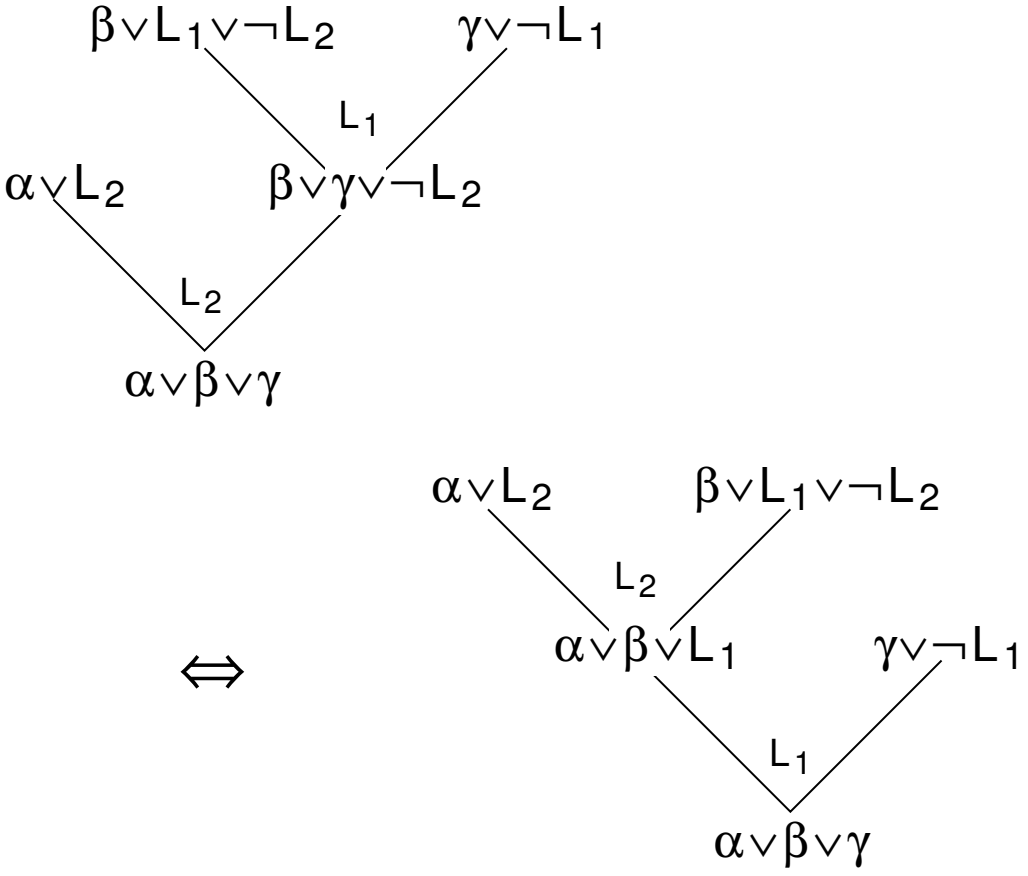


Es darf nur über ein Literal gleichzeitig resolviert werden.

Syntaktische Schlussfolgerungsverfahren

Beispiel (reguläre Resolution).

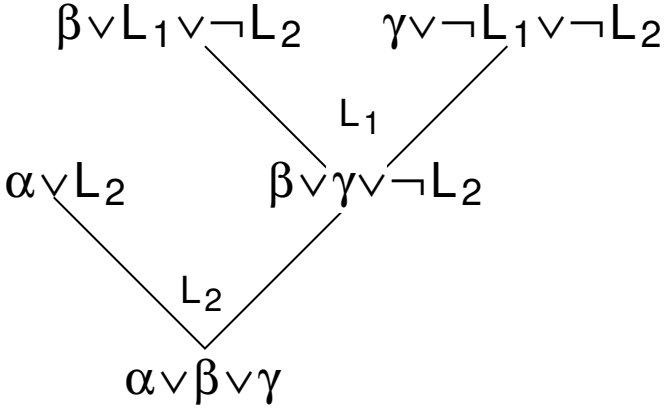
α, β, γ sind Klauseln ohne die Literale $L_1, \neg L_1, L_2, \neg L_2$.



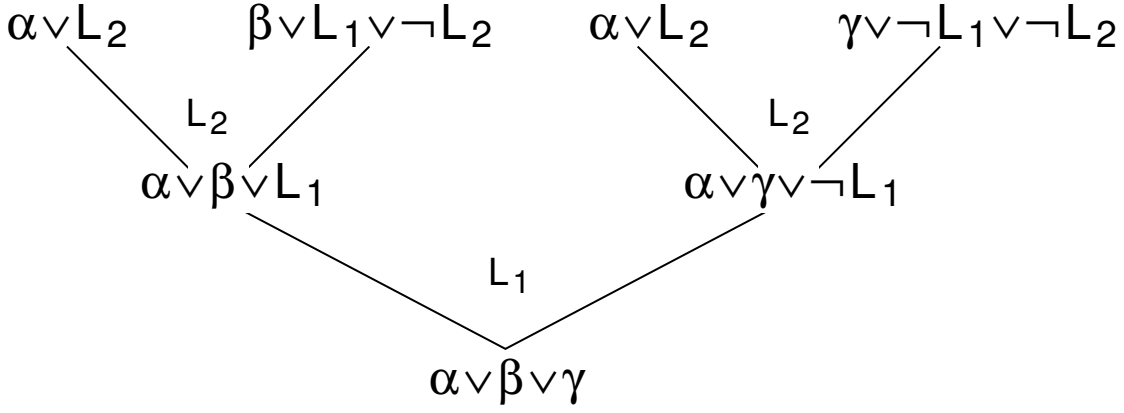
Syntaktische Schlussfolgerungsverfahren

Beispiel (reguläre Resolution).

α, β, γ sind Klauseln ohne die Literale $L_1, \neg L_1, L_2, \neg L_2$.



$\Leftrightarrow$



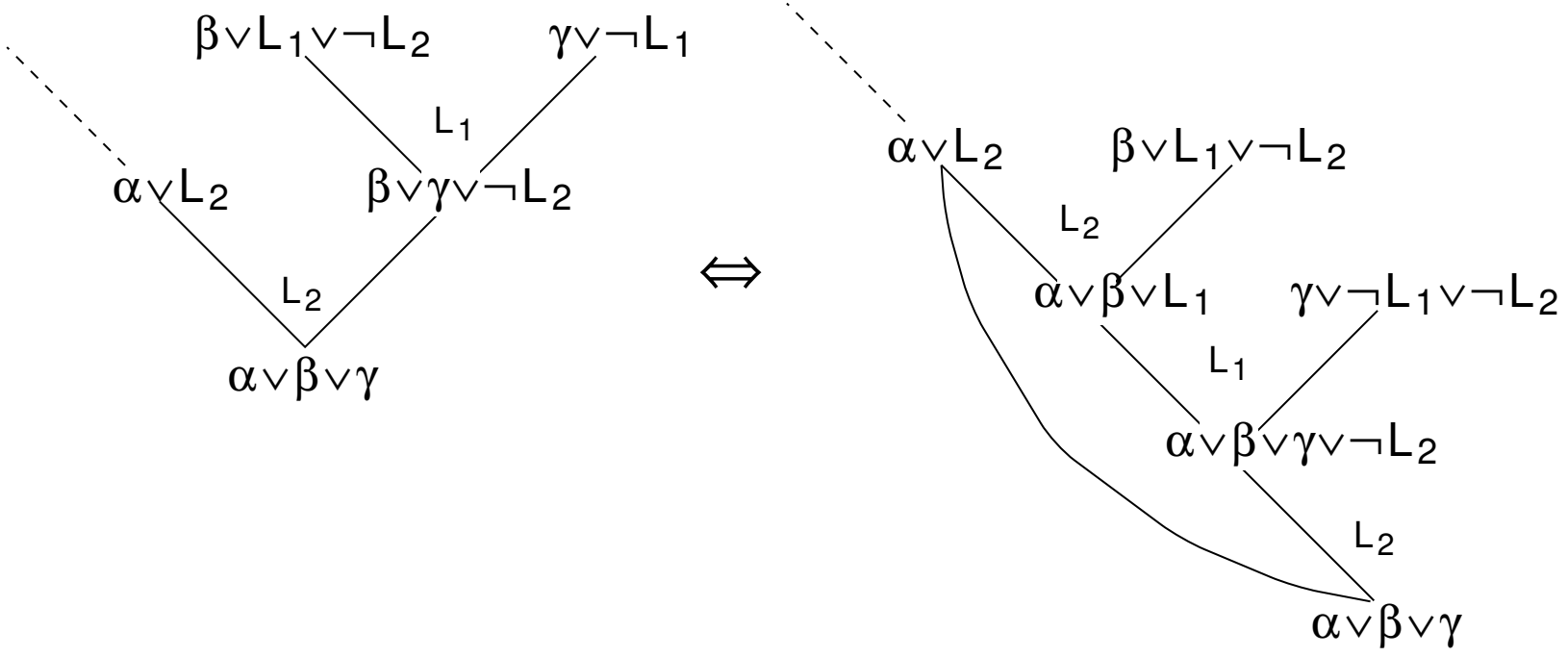
Bemerkungen:

- Die beiden Beispiele zeigen die einzigen Möglichkeiten, über zwei Resolutionsschritte zur Klausel $\alpha \vee \beta \vee \gamma$ zu gelangen.
- Offensichtlich verhindert eine Änderung der Reihenfolge der Literale, über die resolviert wird, keine Ableitung. Also darf man Literale nach einem beliebigen Schema auswählen.
- Verschiedene Herleitungen derselben Klausel können unterschiedlich aufwendig sein.

Syntaktische Schlussfolgerungsverfahren

Beispiel (lineare Resolution immer möglich).

α, β, γ sind Klauseln ohne die Literale $L_1, \neg L_1, L_2, \neg L_2$.



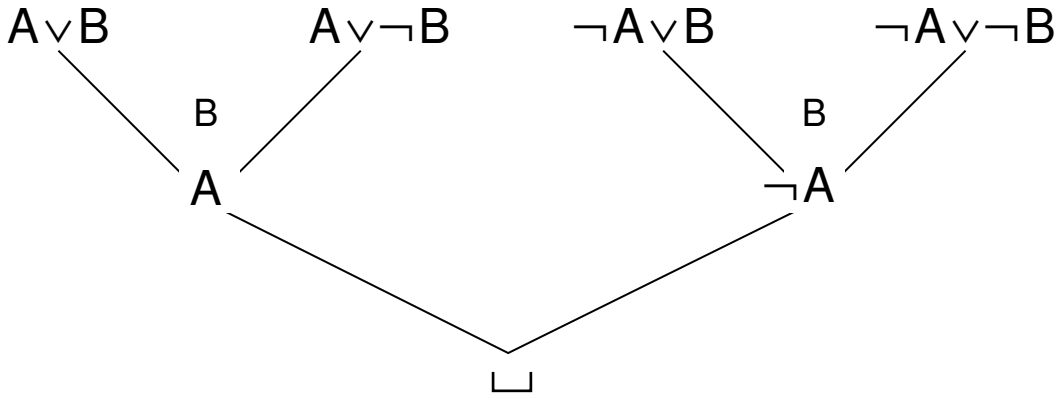
Bemerkungen:

- Der Herleitungsbaum lässt sich immer in eine kanonische Form überführen, der zu einer Seite „keine Äste“ hat.

Syntaktische Schlussfolgerungsverfahren

Beispiel.

$\alpha = \{A \vee B, A \vee \neg B, \neg A \vee B, \neg A \vee \neg B\}$



Syntaktische Schlussfolgerungsverfahren

Satz 3 (Syntax und Semantik im Resolutionskalkül)

1. Der Resolutionskalkül ist korrekt. Sei $\alpha \in$ KNF, π eine Klausel. Dann gilt:

$$\alpha \vdash_{Res} \pi \quad \Rightarrow \quad \alpha \models \pi$$

Syntaktische Schlussfolgerungsverfahren

Satz 3 (Syntax und Semantik im Resolutionskalkül)

1. Der Resolutionskalkül ist korrekt. Sei $\alpha \in$ KNF, π eine Klausel. Dann gilt:

$$\alpha \vdash_{Res} \pi \quad \Rightarrow \quad \alpha \models \pi$$

2. Der Resolutionskalkül ist nicht vollständig. Es gibt ein $\alpha \in$ KNF und eine Klausel π, so dass gilt:

$$\alpha \models \pi \quad \text{aber} \quad \alpha \not\vdash_{Res} \pi$$

Syntaktische Schlussfolgerungsverfahren

Satz 3 (Syntax und Semantik im Resolutionskalkül)

1. Der Resolutionskalkül ist korrekt. Sei $\alpha \in$ KNF, π eine Klausel. Dann gilt:

$$\alpha \vdash_{Res} \pi \quad \Rightarrow \quad \alpha \models \pi$$

2. Der Resolutionskalkül ist nicht vollständig. Es gibt ein $\alpha \in$ KNF und eine Klausel π, so dass gilt:

$$\alpha \models \pi \quad \text{aber} \quad \alpha \not\vdash_{Res} \pi$$

3. Der Resolutionskalkül ist widerlegungsvollständig. Sei $\alpha \in$ KNF. Dann gilt:

$$\alpha \text{ widerspruchsvoll} \quad \Rightarrow \quad \alpha \vdash_{Res} \square$$

Bemerkungen:

- Es gilt sogar die Äquivalenz: α widerspruchsvoll $\quad\Leftrightarrow\quad \alpha \vdash_{Res} \square$

Syntaktische Schlussfolgerungsverfahren

Beweisidee

Zu 1) Korrektheit.

Induktion über die Länge der Herleitung. Im Induktionsanfang zeigt man:

$$\pi_1, \pi_2 \left|\frac{1}{Res}\right. \pi \quad \Rightarrow \quad \pi_1, \pi_2 \models \pi$$

Zu 2) Unvollständigkeit.

Sei $\alpha = A,\ \pi = A \vee B$. Es gilt:

$$\alpha \models \pi \quad \text{aber} \quad \alpha \left|\not\frac{}{Res}\right. \pi$$

Zu 3) Widerlegungsvollständigkeit.

Induktion über die Anzahl der Atome in α.

Bemerkung zur Unvollständigkeit:

Der Resolutionskalkül ermöglicht nicht die Einführung neuer Zeichen. Beachte, dass beim Hilbert-Kalkül (MP + entsprechende Axiome) die Einführung neuer Zeichen durch Axiome wie „$\varphi \rightarrow \varphi \vee \psi$“ realisiert ist.

Syntaktische Schlussfolgerungsverfahren

Satz 4 (Laufzeit des Resolutionskalküls)

Der Resolutionskalkül ist exponentiell. D. h., es gibt Formeln $(\varphi_n), n \in \mathbf{N}$, und eine Konstante $c > 1$, so dass für genügend große n jede Resolutionswiderlegung (Herleitung der leeren Klausel) von φ_n mindestens c^n Klauseln enthält.

[Haken 1985]

Syntaktische Schlussfolgerungsverfahren

Satz 4 (Laufzeit des Resolutionskalküls)

Der Resolutionskalkül ist exponentiell. D. h., es gibt Formeln $(\varphi_n), n \in \mathbf{N}$, und eine Konstante $c > 1$, so dass für genügend große n jede Resolutionswiderlegung (Herleitung der leeren Klausel) von φ_n mindestens c^n Klauseln enthält.

[Haken 1985]

Beweisidee

Betrachte die Pigeon-Hole-Formeln φ_n. Gegeben sind $n+1$ Tauben, n Löcher, und in jedes Loch passt höchstens eine Taube.

Semantik von $X_{i,k}$: Taube i sitzt in Loch k.

a) Klauseln für „Jede Taube sitzt in einem Loch“

$$(X_{1,1} \vee \ldots \vee X_{1,n}) \;\; \ldots \;\; (X_{n+1,1} \vee \ldots \vee X_{n+1,n})$$

b) Klauseln für „In jedem Loch sitzt höchstens eine Taube“

$$\bigwedge_{\substack{1 \le k \le n \\ 1 \le i < j \le n+1}} (\neg X_{i,k} \vee \neg X_{j,k})$$

c) Bilde $\varphi_n \in$ KNF aus Klauseln von (a) und (b). $|\varphi_n| = n \cdot (n+1)^2$

Bemerkungen:

- Die Pigeon-Hole-Formeln sind ein Beispiel für den Versuch, eine injektive Abbildung von einer $(n+1)$-elementigen Menge auf eine n-elementige Menge zu konstruieren.

Syntaktische Schlussfolgerungsverfahren

Die Varianten des Resolutionskalküls können als verschiedene Strategien aufgefasst werden, den Suchraum zu beschränken.

Sie lassen sich in zwei Klassen einteilen, die sich

1. auf die Struktur des Herleitungsbaums bzw.
2. auf semantische Konzepte beziehen.

Definition 26 (Inputklausel)
Eine Klausel der Anfangsformel wird auch Inputklausel genannt.

Syntaktische Schlussfolgerungsverfahren

Zu 1. Varianten in der Struktur des Herleitungsbaums

a) Reguläre Resolution.
   In keiner Resolvente darf ein Literal auftreten, über das bei der Herleitung der Klausel bereits resolviert wurde.

b) Lineare Resolution.
   Für den nächsten Resolutionsschritt wird als eine der Elternklauseln die zuletzt generierte Klausel gewählt. Die andere Klausel ist eine Inputklausel oder eine beliebige, früher generierte Resolvente.

c) Input-Resolution.
   In jedem Resolutionsschritt ist eine Elternklausel eine Inputklausel.

d) Unit-Resolution.
   In jedem Resolutionsschritt ist eine Elternklausel eine Unit-Klausel.

Syntaktische Schlussfolgerungsverfahren

Zu 2. Varianten semantischer Konzepte

a) Semantische Resolution.

Gegeben sei eine Interpretation $\mathcal{I}$. In jedem Resolutionsschritt muss eine Elternklausel mit 0 bewertet sein unter $\mathcal{I}$.

Frage: Was ist, wenn beide Elternklauseln mit 0 bewertet sind?

b) N-Resolution.

Spezialfall der semantischen Resolution mit $\mathcal{I}(A) = 1,\ \forall A \in \Sigma$. In jedem Resolutionsschritt muss eine Elternklausel mit 0 bewertet sein unter $\mathcal{I}$.

Bemerkung: Eine mit 0 bewertete Elternklausel ist eine negative Klausel.

c) P-Resolution.

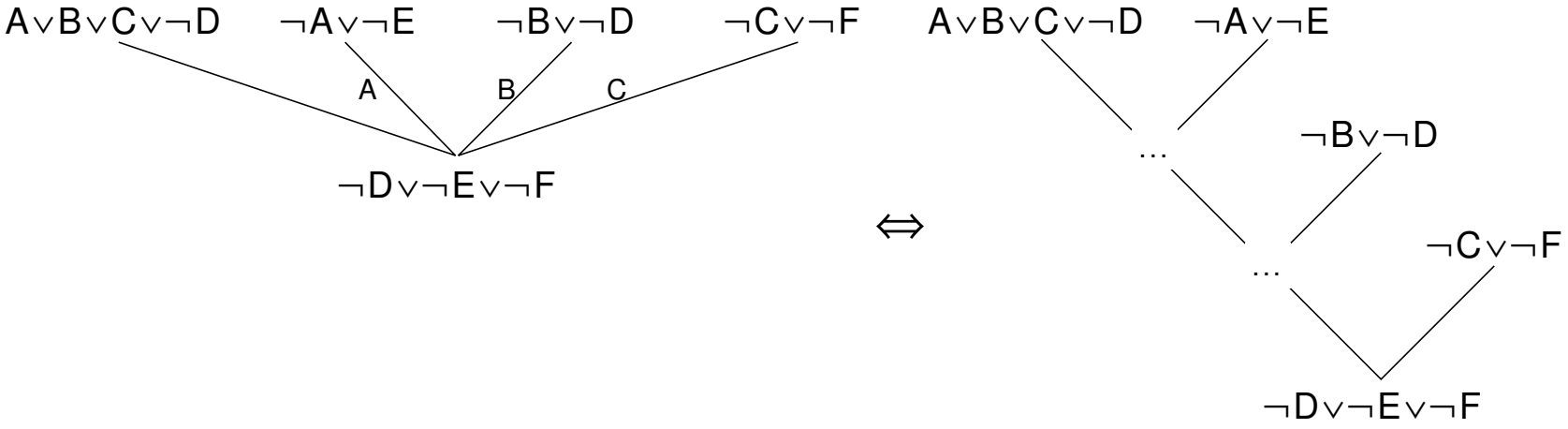
Spezialfall der semantischen Resolution mit $\mathcal{I}(A) = 0,\ \forall A \in \Sigma$. In jedem Resolutionsschritt muss eine Elternklausel mit 0 bewertet sein unter $\mathcal{I}$.

Bemerkung: Eine mit 0 bewertete Elternklausel ist eine positive Klausel.

Syntaktische Schlussfolgerungsverfahren

Zu 2. Varianten semantischer Konzepte (Fortsetzung)

d) (Negative) Hyperresolution.
Ähnlich der N-Resolution. Jedoch werden hier Folgen von N-Reduktionsschritten, die eine negative Resolvente ergeben, zu einem Hyperresolutionsschritt zusammengefasst.



Bemerkungen:

- Zwischen je zwei Klauseln wird höchstens über 1 Literal resolviert.
- Bei der Hyperresolution entstehen weniger Resolventen.

Syntaktische Schlussfolgerungsverfahren

Zu 2. Varianten semantischer Konzepte (Fortsetzung)

e) Set-of-Support-Strategie

Gegeben sein $\alpha_s \subseteq \alpha$ mit α_s erfüllbar (sogenannte Stützmenge). In keinem Resolutionsschritt dürfen beide Elternklauseln aus der Stützmenge stammen.

Bemerkungen:

- Semantische Varianten des Resolutionskalküls stellen natürlich auch syntaktische Schlussfolgerungsverfahren dar. Die Verwendung des Begriffs der Semantik rührt hier daher, dass eine Interpretation $\mathcal{I}$ den Ausgangspunkt spezieller Resolutions-Herleitungen definiert.

Syntaktische Schlussfolgerungsverfahren

Wichtige Resultate für Resolutionskalküle

1. Widerlegungsvollständige Strategien für Formeln $\in$ KNF.
   (a) unbeschränkte Resolution
   (b) reguläre Resolution
   (c) lineare Resolution
   (d) semantische Resolution, N/P-Resolution, Hyperresolution
   (e) Set-of-Support-Strategie

2. Widerlegungsvollständige Strategien für Formeln $\in$ HORN.
   (a) Input-Resolution
   (b) Unit-Resolution

Syntaktische Schlussfolgerungsverfahren

Wichtige Resultate für Resolutionskalküle (Fortsetzung)

3. Äquivalenz von Input-Resolution und Unit-Resolution.

$$\alpha \mid\!\!\overline{}_{Input-Res} \sqcup \quad \Leftrightarrow \quad \alpha \mid\!\!\overline{}_{Unit-Res} \sqcup$$

4. $\alpha \mid\!\!\overline{}_{Unit-Res} \sqcup$ ist in linearer Zeit entscheidbar.

   D. h., es kann in linearer Zeit festgestellt werden, ob sich mit Hilfe der Unit-Resolution aus α die leere Klausel herleiten lässt.
   D. h., wenn $\alpha \in$ Horn, so kann in linearer Zeit festgestellt werden, ob α widerspruchsvoll ist.

Syntaktische Schlussfolgerungsverfahren

Zusammenfassung

Ein Kalkül erweitert eine Logik um den syntaktischen Begriff des Ableitens, der den semantischen Folgerungsbegriff nachbilden soll.

Kalküle in der Logik sind Mechanismen, um Folgerungen zu erzeugen, ohne dass die Folgerungsdefinition (Analyse aller Interpretationen) bemüht werden muss. Ein Kalkül besteht aus einer festen Menge von Schlussregeln.

Die Überprüfung, ob mittels der Schlussregeln eines Kalküls tatsächlich nur Folgerungen erzeugt werden (= Korrektheit) und ob der Kalkül in der Lage ist, alle Folgerungen zu erzeugen (= Vollständigkeit), ist einmal festgestellt worden und braucht nicht weiter in Frage gestellt zu werden.

Deshalb ist eine rein syntaktische Anwendung möglich.

Kapitel L: II (Fortsetzung)

II. Aussagenlogik

- Syntax der Aussagenlogik
- Semantik der Aussagenlogik
- Eigenschaften des Folgerungsbegriffs
- Äquivalenz
- Formeltransformation
- Normalformen

- Bedeutung der Folgerung
- Erfüllbarkeitsalgorithmen
- Semantische Bäume
- Weiterentwicklung semantischer Bäume
- Syntaktische Schlussfolgerungsverfahren
- Erfüllbarkeitsprobleme

Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Die Frage „Gilt $\alpha \models \beta$?“ lässt sich auf einen Erfüllbarkeitstest zurückführen.

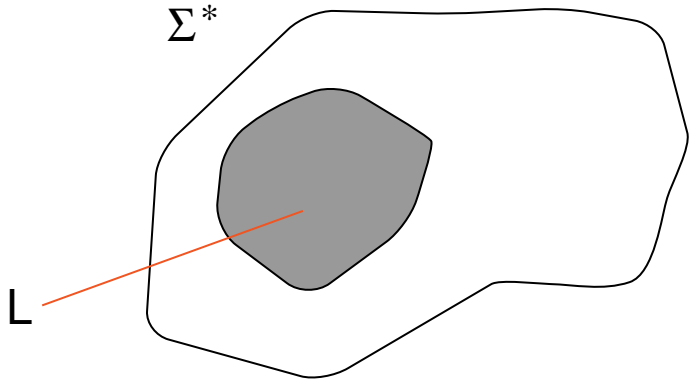
Definition 27 (Komplexitätsklasse P)
Die Komplexitätsklasse P enthält alle Probleme, die sich mit einer deterministischen Turingmaschine in polynomieller Rechenzeit lösen lassen.

Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 28 (Entscheidungsprobleme)
Gegeben sei ein Problem mit der Eingabemenge Σ^*. Dann bezeichnen wir ein Problem als Entscheidungsproblem, wenn für alle $x \in \Sigma^*$ die Antwort (das Ergebnis, die Ausgabe einer Turingmaschine) nur „0“ (nein) oder „1“ (ja) sein kann.

Definition 29 (Sprache)
Gegeben sei ein Entscheidungsproblem mit der Eingabemenge Σ^*. Dann bezeichnen wir diejenige Teilmenge L von Σ^*, deren Elemente die Antwort „1“ haben, als Sprache.



Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 30 (Komplexitätsklasse NP)
Die Komplexitätsklasse NP enthält alle Entscheidungsprobleme, von denen mit einer nichtdeterministischen Turingmaschine M in polynomieller Rechenzeit festgestellt werden kann, dass ein Element zur Sprache des Problems gehört.

Sprachgebrauch: M akzeptiert die Elemente der Sprache (eines Entscheidungsproblems aus NP) in polynomieller Zeit.

Bemerkungen:

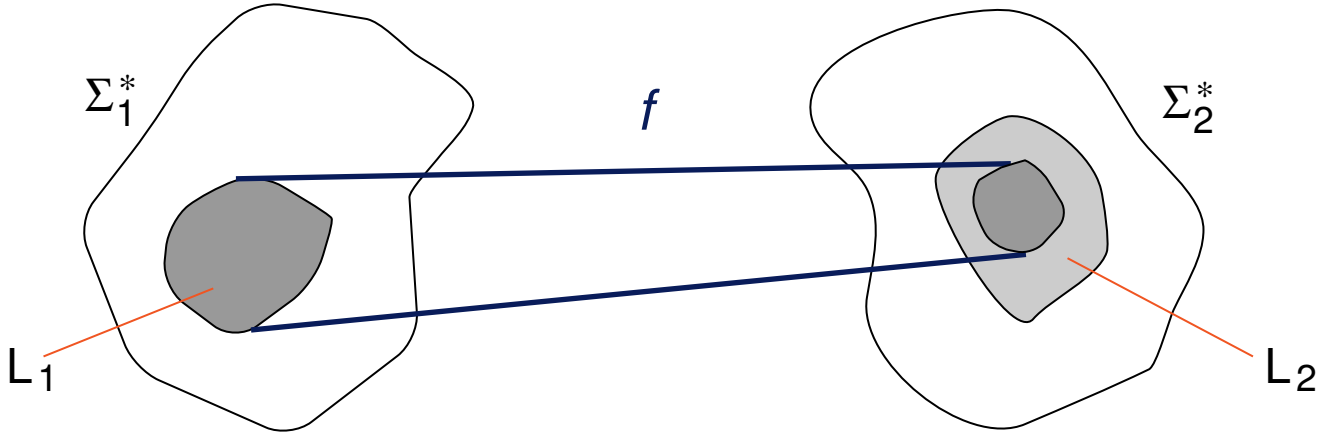
- Probleme aus der Komplexitätsklasse NP sind entscheidbar.
- Vermutung, dass P $\neq$ NP

Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 31 (polynomielle Reduktion)
Eine Sprache $L_1 \subseteq \Sigma_1^*$ lässt sich polynomiell auf eine Sprache $L_2 \subseteq \Sigma_2^*$ reduzieren, in Zeichen: $L_1 \leq L_2$, wenn es eine polynomiell berechenbare Transformation $f : \Sigma_1^* \to \Sigma_2^*$ gibt, so dass gilt:

$$\forall x \in \Sigma_1^* : \quad x \in L_1 \Leftrightarrow f(x) \in L_2$$



Bemerkungen:

- $L_1 \leq L_2$ kann interpretiert werden als: L_1 ist nicht schwerer als L_2.

Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 32 (hart bzgl. einer Menge von Sprachen)
Eine Sprache L heißt hart für eine Menge von Sprachen $\mathcal{L}$, falls sich jede Sprache $L' \in \mathcal{L}$ auf L reduzieren lässt. In Zeichen: $\forall L' \in \mathcal{L} : L' \leq L$.

Definition 33 (vollständig bzgl. einer Menge von Sprachen)
Eine Sprache L heißt vollständig für eine Menge von Sprachen $\mathcal{L}$, falls L hart für $\mathcal{L}$ ist, und falls zusätzlich $L \in \mathcal{L}$ gilt.

Erfüllbarkeitsprobleme

Definition 34 (SAT*)

SAT* = $\{\alpha \mid \alpha$ aussagenlogische Formel $\wedge\ \alpha$ erfüllbar $\}$

Satz 5 (Komplexität von SAT*)

1. SAT* $\in$ NP
2. SAT* ist NP-hart. [Cook 1971]

Beweisidee

Zu (1): Elemente aus SAT* werden von einer nichtdeterministischen Turingmaschine in polynomieller Zeit akzeptiert.

Frage: Wie zeigt man das?

Zu (2): Alle Probleme aus NP lassen sich in polynomieller Zeit auf SAT* reduzieren.

Frage: Wie hat Cook das gezeigt?

Bemerkung: SAT ist NP-vollständig.

Erfüllbarkeitsprobleme

Definition 35 (SAT)

SAT = $\{\alpha \mid \alpha \in \text{KNF} \wedge \alpha \text{ erfüllbar } \}$

Satz 6 (Komplexität von SAT)

SAT ist NP-vollständig.

Beweisidee

Reduktion von SAT* auf SAT, in Zeichen: SAT$^* \leq$ SAT.

Erfüllbarkeitsprobleme

Definition 36 (3SAT)

3SAT = $\{\alpha \mid \alpha \in$ 3KNF $\wedge\ \alpha$ erfüllbar $\}$

Satz 7 (Komplexität von 3SAT)

3SAT ist NP-vollständig.

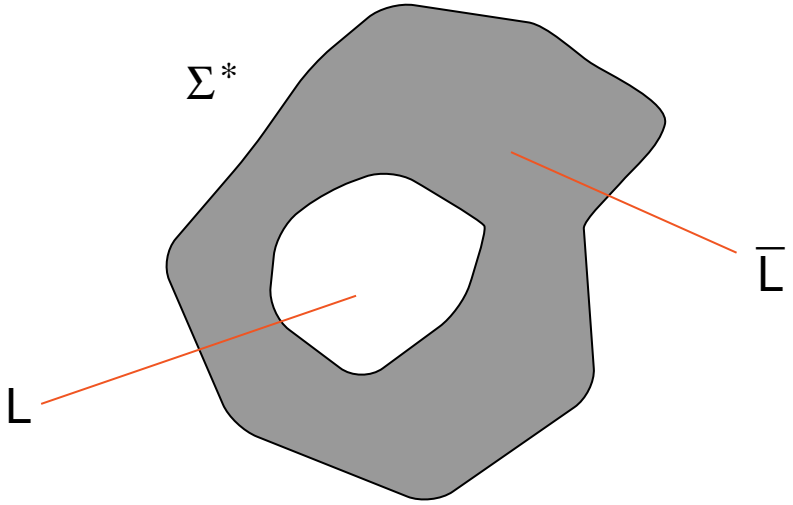
Beweisidee

Reduktion von SAT auf 3SAT, in Zeichen: SAT $\leq$ 3SAT.

Erfüllbarkeitsprobleme

Wiederholung (theoretische Informatik)

Definition 37 (Komplexitätsklasse co-NP)
Die Komplexitätsklasse co-NP enthält alle Entscheidungsprobleme, deren Komplementsprachen $\overline{L}$, $\overline{L} := \Sigma^* \setminus L$, in NP liegen.



Vermutung: NP $\neq$ co-NP

Erfüllbarkeitsprobleme

Definition 38 (DEDUCT)

DEDUCT = $\{(\alpha, L) \mid \alpha \in$ KNF $\wedge\ L$ Literal mit $\alpha \models L\}$

Satz 8 (Komplexität von DEDUCT)

DEDUCT ist co-NP-vollständig.

Beweis 3

Reduktion von SAT auf $\overline{\text{DEDUCT}}$.

- $\overline{\text{DEDUCT}} = \{(\alpha, L) \mid \alpha \in$ KNF $\wedge\ L$ Literal mit $\alpha \not\models L\}$
- Sei A ein Atom mit $A \notin \mathit{atoms}(\alpha)$.
- $\underbrace{\alpha \in}_{x}$ KNF beliebig. $\underbrace{(\alpha, A)}_{f(x)}$ eine spezielle Instanz des Entscheidungsproblems.
- Es gilt: α erfüllbar $\Leftrightarrow\ \alpha \not\models A$
  Bzw.: $x \in$ SAT $\Leftrightarrow\ f(x) \in \overline{\text{DEDUCT}}$
- Die Komplementsprache von DEDUCT ist $\overline{\text{DEDUCT}}$ und ist NP-vollständig. Also ist DEDUCT co-NP-vollständig.

Erfüllbarkeitsprobleme

Definition 39 (EQUIV)

EQUIV = $\{(\alpha, \beta) \mid \alpha, \beta \in \mathsf{KNF} \wedge \alpha \approx \beta\}$

Satz 9 (Komplexität von EQUIV)

EQUIV ist co-NP-vollständig.

Beweis 4

Reduktion von $\overline{\mathsf{SAT}}$ auf EQUIV.

Rest als Übungsaufgabe.

Erfüllbarkeitsprobleme

Definition 40 (2SAT)

2SAT = $\{\alpha \mid \alpha \in$ 2KNF $\wedge\ \alpha$ erfüllbar $\}$

Satz 10 (Komplexität von 2SAT)

2SAT $\in$ P. [Aspvall 1980]

Erfüllbarkeitsprobleme

Beweisidee (Komplexität von 2SAT)

1. Units L durch $L \vee L$ ersetzen $\Rightarrow$ alle Klauseln haben genau zwei Literale.
2. Generierung eines gerichteten Graphen $G = \langle V, E \rangle$. V enthält alle Literale aus α sowie deren Komplemente.
3. Aus jeder Klausel (L_1, L_2) werden zwei Kanten. Es gilt:
   $(L_1, L_2) \approx (L_1 \vee L_2) \wedge (L_2 \vee L_1) \approx (\neg\neg L_1 \vee L_2) \wedge (\neg\neg L_2 \vee L_1) \approx (\neg L_1 \rightarrow L_2) \wedge (\neg L_2 \rightarrow L_1)$
4. Die starken Zusammenhangskomponenten von G sind zyklische Ketten von Implikationen. Sie können nur dann erfüllt sein, wenn alle beteiligten Literale entweder mit 0 oder mit 1 bewertet sind. Folglich dürfen alle starken Zusammenhangskomponenten zu einem Knoten kontrahiert werden.
5. Erzeugung einer Initialbewertung: Bewertung aller Knoten, die nur ausgehende Kanten haben, mit 0. Bewertung aller Knoten, die nur eingehende Kanten haben, mit 1. (Least-Commitment-Prinzip)
6. Propagierung der Initialbewertung entlang einer topologischen Sortierung.
7. α ist erfüllbar $\Leftrightarrow$ keine starke Zusammenhangskomponente enthält ein Literal als positive und negative Instanz.

Erfüllbarkeitsprobleme

Definition 41 (SAT-Probleme in HORN)

1. SAT∩HORN = $\{\alpha \mid \alpha \in$ HORN $\wedge\ \alpha$ erfüllbar $\}$
2. SAT∩DHORN = $\{\alpha \mid \alpha \in$ DHORN $\wedge\ \alpha$ erfüllbar $\}$
3. DEDUCT∩HORN = $\{(\alpha, L) \mid \alpha \in$ HORN $\wedge\ L$ Literal mit $\alpha \models L\}$
4. EQUIV∩HORN = $\{(\alpha, \beta) \mid \alpha, \beta \in$ HORN $\wedge\ \alpha \approx \beta\}$

Satz 11 (Komplexität von SAT-Problemen in HORN)

Die Probleme SAT∩HORN, SAT∩DHORN, DEDUCT∩HORN und EQUIV∩HORN sind in P.

Erfüllbarkeitsprobleme