Aussagenlogische Kalküle

Ziel:
mit Hilfe von schematischen *Regeln* sollen *alle* aus einer Formel logisch folgerbaren Formeln durch (prinzipiell syntaktische) Umformungen abgeleitet werden können.

Derartige Systeme von "Beweisregeln" werden *Kalküle* genannt.

Beispiel einer Beweisregel:

Die Regel mit Schemavariablen α,β,γ

$$\frac{\alpha \vee \beta, \beta \to \gamma, \gamma \to \alpha}{\alpha}$$

Eine Instanz mit

$$\begin{array}{rcl} \alpha & = & \neg P_1 \\ \beta & = & \mathbf{0} \\ \gamma & = & P_1 \wedge P_2 \end{array}$$

$$\frac{\neg P_1 \vee \mathbf{0}, \mathbf{0} \to (P_1 \wedge P_2), (P_1 \wedge P_2) \to \neg P_1}{\neg P_1}$$

Definition für Beweisregeln, präzisiert:

Für ein $n \in N$ ist eine *n-stellige Regel* R eine entscheidbare, $n+1$-stellige Relation über der Menge der Formeln.
Ist $(u_1, \ldots, u_n, u_{n+1}) \in R$, so heißen

- $(u_1, \ldots, u_n, u_{n+1})$ eine *Instanz* von R
- $u_1, \ldots, u_n$ die *Prämissen* dieser Instanz
- u_{n+1} die *Conclusio* dieser Instanz.

Wir schreiben meist

$$\frac{u_1, \ldots, u_n}{u_{n+1}}$$

Die Instanzen von nullstelligen Regeln heissen auch *Axiome*.

Ableitungen in einem Kalkül, abstrakt:

Sei eine Formelmenge L, eine Menge M von Voraussetzungen und eine Menge von Regeln festgelegt.

Definition

Eine *Ableitung* aus M ist eine Folge $(u_1, \ldots, u_m)$ von Formeln in L, so dass für jedes $i \in \{1, \ldots, m\}$ gilt:

- u_i ist Axiom; oder
- $u_i \in M$; oder
- es gibt eine Instanz
$$\frac{u_{j_1}, \ldots, u_{j_n}}{u_i}$$
einer Regel mit $j_1, \ldots, j_n < i$

Man beachte, dass u_1 ein Axiom oder ein Element von M sein muss.

Eine Formel A heisst *ableitbar* aus M, kurz

$$M \vdash A$$

genau dann, wenn es eine Ableitung $(u_1, \ldots, u_m)$ gibt mit $u_m = A$.

- Für $\emptyset \vdash u$ schreiben wir $\vdash u$, für $\{v\} \vdash u$ schreiben wir $v \vdash u$.
- Falls erforderlich wird der Kalkül in der Notation mit angezeigt, also $M \vdash_{Kal} u$.

Was sollte ein Kalkül leisten können?

Korrektheit: $M \vdash A \Rightarrow M \models A$

Vollständigkeit: $M \models A \Rightarrow M \vdash A$

Beispiel:

Die Regeln des Hilbertkalküls

Ax1: $\dfrac{}{\alpha \to (\beta \to \alpha)}$

Ax2: $\dfrac{}{(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma))}$

Ax3: $\dfrac{}{(\neg\alpha \to \neg\beta) \to (\beta \to \alpha)}$

Mp: $\dfrac{\alpha, \alpha \to \beta}{\beta}$ (Modus ponens)

Der Hilbertkalkül ist aber für das automatische Beweisen nicht so gut geeignet.

Der Resolutionskalkül

wird verwendet, um die *Unerfüllbarkeit* einer endlichen Menge aussagenlogischer Formeln zu beweisen.

- Zur Überprüfung, ob F eine Tautologie darstellt, kann man $\neg F$ auf Unerfüllbarkeit testen.
- Eine Formel G **folgt** aus einer Formelmenge $\{F_1, \ldots, F_k\}$ klassisch gdw. $F_1 \wedge \ldots \wedge F_k \wedge \neg G$ unerfüllbar ist: **Widerlegungsverfahren** („Refutation").

Exkurs: Unerfüllbarkeit und logische Folgerung

$$\Gamma \cup \{\neg F\} \text{ ist unerfüllbar gdw. } \Gamma \models F$$

Modelltheoretische Argumentation:

"$\rightarrow$"

Wenn $\Delta \models \varphi$, dann sind alle Modelle für Δ auch Modelle für φ. Damit ist keine dieser Interpretationen Modell für $\neg\varphi$ und damit ist $\Delta \cup \neg\varphi$ unerfüllbar.

"$\leftarrow$"

Sei nun $\Delta \cup \neg\varphi$ unerfüllbar, aber Δ erfüllbar. Sei I die Interpretation, die Δ erfüllt. I erfüllt nicht $\neg\varphi$, denn sonst wäre $\Delta \cup \neg\varphi$ erfüllbar. Damit folgt, dass I φ erfüllt. (Eine Interpretation muss entweder φ oder $\neg\varphi$ erfüllen.). Da dies für willkürliche Modelle I für Δ gilt, gilt es für alle I, die Δ erfüllen. Damit sind alle Modelle für Δ auch Modelle für φ und φ ist logische Konsequenz von Δ.

Resolutionskalkül: beruht auf der schematischen, aussagenlogischen Schlussregel der Resolution.

Def. "Resolvente":

a) Seien K_1, K_2 und K_3 Klauseln. Die Klausel K_3 heißt eine **Resolvente** von K_1 und K_2, wenn es

ein Literal L gibt, so dass die folgenden beiden Bedingungen erfüllt sind:

1) Es gilt $L \in K_1$ und $\neg L \in K_2$. Ist hierbei $L = \neg A$ ein negatives Literal, so sei $\neg L = A$.

2) $K_3 = (K_1 \setminus \{L\}) \cup (K_2 \setminus \{\neg L\})$

b) Ist K_3 eine Resolvente von K_1 und K_2, so verwenden wir nebenstehende grafische Darstellung:

$$\begin{tikzcd} K_1 \arrow[dr, dash] & & K_2 \arrow[dl, dash] \\ & K_3 & \end{tikzcd}$$

c) Dabei ist die leere Klausel $K_3 = \square$ zulässig. Sie ergibt sich z.B. als Resolvente von $K_1 = \{L\}$ und $K_2 = \{\neg L\}$.

Eine Klauselmenge, die $\square$ enthält, wird als **unerfüllbar** bezeichnet.

Merkmale des Resolutionskalküls

- Widerlegungskalkül
- Voraussetzung:
  Alle Formeln sind in konjunktiver Normalform.
- Es gibt eine einzige Regel: die Resolutionsregel eine Modifikation des Modus ponens.

- es sind keine logischen Axiome notwendig.

Syntax:

Eine *Klausel* ist eine endliche Menge von Literalen.

$\square$ ist die *leere Klausel*.

Eine Klausel

$$\{P_1, P_2\} \quad \{P_1, \neg P_2\}$$

lesen wir als die Disjunktion der ihr angehörenden Literale:

$$P_1 \vee P_2 \quad P_1 \vee \neg P_2$$

Die grundlegende Datenstruktur, auf der wir operieren, sind die *Mengen von Klauseln*,

$$\{\{P_1, P_2\}, \{P_1, \neg P_2\}, \{\neg P_1, P_2\}, \{\neg P_1, \neg P_2\}\}$$

gelesen als Konjunktion der Klauseln.

$$(P_1 \vee P_2) \wedge (P_1 \vee \neg P_2) \wedge (\neg P_1 \vee P_2) \wedge (\neg P_1 \vee \neg P_2)$$

Bis auf die Reihenfolge der Disjunktions- bzw. Konjunktionsglieder sind also in umkehrbar eindeutiger Weise die Formeln in konjunktiver Normalform als Klauselmengen wiedergegeben.

Die Mengenschreibweise bringt Vorteile hinsichtlich der Ausnutzung der Assoziativität und Kommutativität von Konjunktion und Disjunktion.

Semantik:

Ist $I : \Sigma \to \{W, F\}$ eine Interpretation der Atome, so hat man als zugehörige Auswertung
val_I : Mengen von Klauseln $\to \{W, F\}$:

$$val_I(M) = \begin{cases} W & \text{falls für alle } C \in M \text{ gilt:} \\ & val_I(\{C\}) = W \\ F & \text{sonst} \end{cases}$$

$$val_I(\{C\}) = \begin{cases} W & \text{falls ein } L \in C \text{ existiert mit} \\ & I(L) = W \\ F & \text{sonst} \end{cases}$$

für Klauseln C,
Es folgt also

$$val_I(\{\Box\}) = F,$$
$$val_I(\emptyset) = W$$

($\emptyset$ ist die leere Klauselmenge.)

Die Regel:

Die *aussagenlogische Resolutionregel* ist die Regel

$$\frac{C_1 \cup \{P\}, C_2 \cup \{\neg P\}}{C_1 \cup C_2}$$

mit einer AL-Variablen P und Klauseln C_1, C_2
$C_1 \cup C_2$ heisst *Resolvente* von $C_1 \cup \{P\}$, $C_2 \cup \{\neg P\}$.
Der Resolutionskalkül enthält die Resolutionsregel als einzige Regel.

Ziel des Resolutionskalküls ist der Nachweis der Unerfüllbarkeit einer Klauselmenge K durch Ableitung von $\square$ aus K. Wünschenswert sind dabei:

Korrektheit: $\square$ ist nur dann ableitbar aus K, wenn K unerfüllbar ist.

Vollständigkeit: Wenn K unerfüllbar ist, so lässt sich $\square$ aus K ableiten.

Beispiel:

Gegeben sei die Klauselmenge
$M = \{\{P_1, P_2\}, \{P_1, \neg P_2\}, \{\neg P_1, P_2\}, \{\neg P_1, \neg P_2\}\}$

$$\frac{\{P_1, P_2\}, \{P_1, \neg P_2\}}{\{P_1\}}$$

$$\frac{\{\neg P_1, P_2\}, \{\neg P_1, \neg P_2\}}{\{\neg P_1\}}$$

$$\frac{\{P_1\}, \{\neg P_1\}}{\square}$$

Insgesamt:

$$M \vdash_{Res} \square$$

Ein zweites Beispiel:

Es soll gezeigt werden, dass

$$(A \to B) \to ((B \to C) \to (A \to C))$$

eine Tautologie ist.
Dazu werden wir zeigen, dass die Negation dieser Formel

$$\neg((A \to B) \to ((B \to C) \to (A \to C)))$$

nicht erfüllbar ist.
Klauselnormalform:

$$M = \{\{\neg A, B\}, \{\neg B, C\}, \{A\}, \{\neg C\}\}$$

$$M = \{\{\neg A, B\}, \{\neg B, C\}, \{A\}, \{\neg C\}\}$$

Ableitung der leeren Klausel aus M:

(1)	[]	$\{\neg A, B\}$
(2)	[]	$\{\neg B, C\}$
(3)	[]	$\{A\}$
(4)	[]	$\{\neg C\}$
(5)	[1, 3]	$\{B\}$
(6)	[2, 5]	$\{C\}$
(7)	[4, 6]	$\square$

Korrektheit und Vollständigkeit:

Theorem

Für eine Menge M von Klauseln gilt

$$M \text{ unerfüllbar} \quad \Leftrightarrow \quad M \vdash_{R0} \square.$$

Beweisidee für die Vollständigkeit ("$\Rightarrow$"):

- Aus $M \nvdash_{R0} \square$ beweisen wir die Erfüllbarkeit von M.
- Beliebige aber feste Reihenfolge der Atome: $P_0, \ldots, P_n, \ldots$
- M_0 sei die Menge aller Klauseln C mit $M \vdash_{R0} C$.
  Es gilt also $M \subseteq M_0$ und $\square \notin M_0$.
- Wir definieren (induktiv) eine Interpretation I so dass:
  für alle $C \in M_0$ gilt $val_I(C) = W$.
  Insbesondere ist dann M als erfüllbar nachgewiesen.

Umformulierung des Theorems:

- **Definition**:

  a) Für jede Klauselmenge K setzen wir

  $\mathrm{Res}^1(K) = K \cup \{R \mid R \text{ Resolvente zweier Klauseln in } K\}$.

  b) Für $n \geq 2$ sei $\mathrm{Res}^n(K) = \mathrm{Res}(\mathrm{Res}^{n-1}(K))$.

  Für $n = 0$ sei $\mathrm{Res}^0(K) = K$. Wir nennen $\mathrm{Res}^n(K)$ die Menge der Resolventen **n-ter Stufe** von K.

  c) Schließlich setzen wir $\mathrm{Res}^\infty(K) = \bigcup_{n \geq 0} \mathrm{Res}^n(K)$.

- **Satz**: (Resolutionssatz der Aussagenlogik)

  Eine Formel F ist genau dann unerfüllbar, wenn $\square \in \mathrm{Res}^\infty(K(F))$ gilt.

- **Algorithmus**: Erfüllbarkeitstest für aussagenlogische Formeln

  Gegeben sei eine aussagenlogische Formel F in KNF.

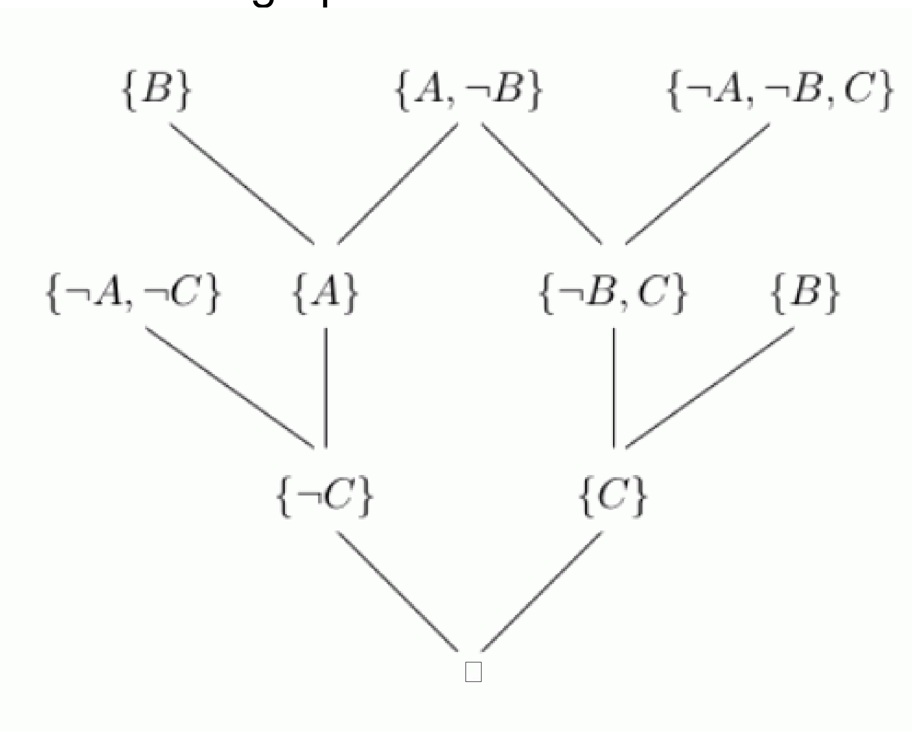
  1) Bilde die Klauselmenge K(F).
  2) Für n = 1, 2, 3, . . . berechne Res^n (K(F)) solange, bis

     $\square \in Res^n(K(F))$ oder $Res^n(K(F)) = Res^{n-1}(K(F))$ gilt.
  3) Im ersten Fall gib "F ist unerfüllbar" aus,
     im zweiten Fall gib "F ist erfüllbar" aus und stoppe.

- **Definition**:
  Eine **Deduktion** (**Herleitung**) der leeren Klausel aus einer Klauselmenge F ist eine Folge $K_1, \ldots, K_m$ von Klauseln mit $K_m = \square$ und jedes K_i (i = 1,...,m) ist Element aus F oder kann aus Klauseln K_a, K_b mit a, b < i resolviert werden.

- Beispiel: **K** = {{B}, {A, ¬B}, {¬A, ¬B, C}, {¬A, ¬C}}

  Resolutionsgraph:



Das Ableiten der leeren Klausel aus einer Menge von Klauseln kann als *Suchproblem* aufgefasst werden.

Es sind verschiedene Suchstrategien möglich, z.B.:

- "linear": Die Resolvente ist jeweils wieder Elternklausel im nächsten Schritt
- "input resolution": eine der Elternklauseln ist immer eine Eingabeklausel
- "unit resolution": eine der Elternklauseln darf nur ein Literal enthalten
- "ordered resolution"
- SLD-Resolution (s.u.)

1-Resolution

Die 1-Resolutionsregel ist ein Spezialfall der allgemeinen Resolutionsregel:

$$\frac{\{P\}, C_2 \cup \{\neg P\}}{C_2} \qquad \frac{\{\neg P\}, C_2 \cup \{P\}}{C_2}$$

Der 1-Resolutionskalkül ist nicht vollständig.
Die Klauselmenge

$$E = \{\{P_1, P_2\}, \{P_1, \neg P_2\}, \{\neg P_1, P_2\}, \{\neg P_1, \neg P_2\}\}$$

ist nicht erfüllbar, aber mit 1-Resolution ist aus E nichts ableitbar, also auch nicht $\square$

Lineare Resolution

- **Definition**:

  Sei F eine aussagenlogische Formel in KNF.

  a) Die leere Klausel ist aus K(F) **linear resolvierbar**, falls es eine Klausel $K_0 \in K(F)$ gibt und eine Folge von Klauseln $K_1, \ldots, K_n$ mit folgenden Eigenschaften:

  a1) für $i = 1, \ldots, n$ gilt

$$\begin{array}{ccc} K_{i-1} & & B_{i-1} \\ & \searrow \quad \swarrow & \\ & K_i & \end{array}$$

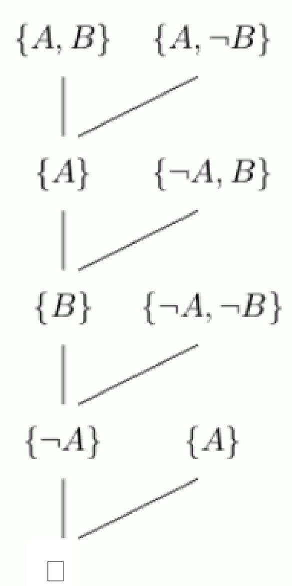
  wobei die **Seitenklausel** B_{i-1} entweder ein Element von K(F) ist oder $B_{i-1} = K_j$ mit $j < i - 1$.

  a2) $K_n = \square$.

  b) Eine Folge von Klauseln $K_0, \ldots, K_n$ wie in a) heißt eine **lineare Resolution** von F.

Beispiel: Sei K = {{A, B}, {A, ¬B}, {¬A, B}, {¬A, ¬B}}

Lineare Resolution



Input-Resolution

- **Definition**:
  Sei F eine aussagenlogische Formel in KNF.
  Eine **Input-Resolution** von F ist eine lineare Resolution von F, bei der in jedem Schritt als Seitenklausel eine der Klauseln aus K(F) verwendet wird.

- Bemerkung:
  Nicht jede unerfüllbare aussagenlogische Formel in KNF besitzt eine Input-Resolution;
  u.a. die im letzten Beispiel.

SLD-Resolution

- **Definition**:
  Sei F eine Hornformel.
  Eine **SLD-Resolution** ("Linear resolution with Selection function restricted to Definite clauses") ist eine Input-Resolution der folgenden Form:
  a) K_0 ist eine negative Klausel, die sog. **Zielklausel**.
  b) Bei jedem Resolutionsschritt ist eine der Elternklauseln eine nicht-negative Hornklausel, also eine **Programmklausel**.

- Anmerkungen:
  - Eine SLD-Resolution benötigt immer auch eine **Auswahlfunktion**, die bei jedem Schritt angibt, welches Literal als nächstes zu resolvieren ist.
    Diese ist bei der Logik-Programmierung zu spezifizieren.

- **Satz**:
  1. Die lineare Resolution ist vollständig,
     d.h. für jede unerfüllbare Klauselmenge K gibt es eine Klausel $k \in K$, so dass die leere Klausel durch lineare Resolution aus K, basierend auf k, herleitbar ist.
  2. Die SLD-Resolution ist vollständig für Hornformeln.

Die SLD-Resolution ist die Grundlage des logischen Schießens in der Programmiersprache Prolog.

Vorgriff: *Prolog-Syntax*

Tatsachenklauseln (atomare Formeln, "Fakten") werden in der Form **`A:-.`** oder **`A.`** eingegeben.

Prozedurklauseln (Klauseln mit genau einem positiven und mindestens einem negativen Literal) werden in der Form

$$\mathtt{A:-B_1,B_2,\ldots,B_{\mathit{r}}}$$

eingegeben. Dies bedeutet

$$A \vee \neg B_1 \vee \ldots \vee \neg B_r$$

oder, gleichwertig:

$$(B_1 \wedge \ldots \wedge B_r) \rightarrow A\ .$$

Der Tableaukalkül

Wesentliche Eigenschaften

- Widerlegungskalkül: Testet auf Unerfüllbarkeit

$$M \models A \quad \Leftrightarrow \quad M \cup \{\neg A\} \vdash_{\mathbf{T0}} \mathbf{0}.$$

- Beweis durch Fallunterscheidung
- Top-down-Analyse der gegebenen Formeln

Vorteile

- Intuitiver als Resolution
- Formeln müssen nicht in Normalform sein
- Falls Formelmenge erfüllbar ist (Test schlägt fehl), wird ein Gegenbeispiel (eine erfüllende Interpretation) konstruiert

Nachteil

- Mehr als eine Regel

Der Tableaukalkül verwendet eine Erweiterung der aussagenlogischen Formeln als Hilfskonstruktion: "Vorzeichenformeln"

Definition (Vorzeichenformel)

Eine **Vorzeichenformel** ist eine Zeichenkette der Gestalt

$$0A \text{ oder } 1A \qquad \text{mit } A \in For0.$$

0, 1 sind neue Sonderzeichen (die Vorzeichen) im Alphabet der Objektsprache.

Definition

Wir setzen val_I fort auf die Menge aller Vorzeichenformeln durch

$$val_I(0A) = val_I(\neg A),$$

und

$$val_I(1A) = val_I(A).$$

2 Typen von Vorzeichenformeln:

Konjunktive Formeln: Typ α

- $1(A \wedge B)$
- $0(A \vee B)$
- $0(A \rightarrow B)$
- $0\neg A$
- $1\neg A$

Disjunktive Formeln: Typ β

- $0(A \wedge B)$
- $1(A \vee B)$
- $1(A \rightarrow B)$

Zuordnungsregeln Formeln / Unterformeln

α	α_1	α_2
$1(A \wedge B)$	$1A$	$1B$
$0(A \vee B)$	$0A$	$0B$
$0(A \rightarrow B)$	$1A$	$0B$
$0\neg A$	$1A$	$1A$
$1\neg A$	$0A$	$0A$

β	β_1	β_2
$0(A \wedge B)$	$0A$	$0B$
$1(A \vee B)$	$1A$	$1B$
$1(A \rightarrow B)$	$0A$	$1B$

Regeln des Tableaukalküls:

$\dfrac{\alpha}{\begin{array}{c}\alpha_1\\ \alpha_2\end{array}}$	konjunktiv	$\begin{array}{c}1(p \wedge q)\\ \vert\\ 1p\\ \vert\\ 1q\end{array}$
$\dfrac{\beta}{\beta_1 \mid \beta_2}$	disjunktiv	$\begin{array}{ccc} & 1(p \vee q) & \\ / & & \backslash \\ 1p & & 1q\end{array}$
$\dfrac{\begin{array}{c}1F\\ 0F\end{array}}{*} \quad \dfrac{\mathbf{01}}{*} \quad \dfrac{\mathbf{10}}{*}$	Widerspruch	$\begin{array}{c}1F\\ \vert\\ 0F\\ \vert\\ *\end{array} \quad \begin{array}{c}\mathbf{01}\\ \vert\\ *\end{array} \quad \begin{array}{c}\mathbf{10}\\ \vert\\ *\end{array}$

Beachte: Disjunktionen führen zu Verzweigungen.

Instanzen der α- und β-Regel:

Instanzen der α-Regel

$$\dfrac{1(P \wedge Q)}{\begin{array}{c}1P\\ 1Q\end{array}} \qquad \dfrac{0(P \vee Q)}{\begin{array}{c}0P\\ 0Q\end{array}} \qquad \dfrac{0(P \rightarrow Q)}{\begin{array}{c}1P\\ 0Q\end{array}} \qquad \dfrac{0\neg P}{1P} \qquad \dfrac{1\neg P}{0P}$$

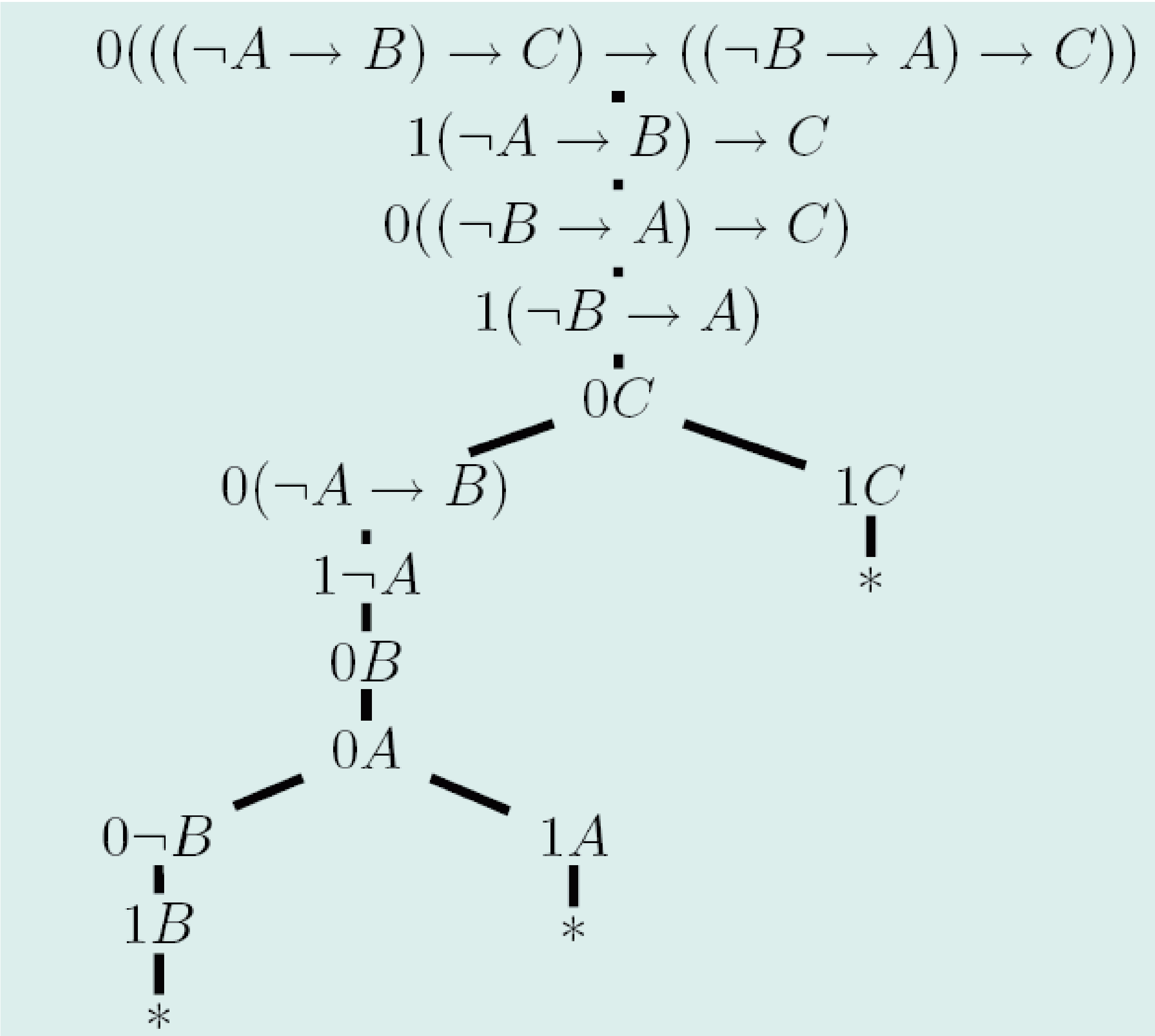
Instanzen der β-Regel

$$\dfrac{1(P \vee Q)}{1P \mid 1Q} \qquad \dfrac{0(P \wedge Q)}{0P \mid 0Q} \qquad \dfrac{1(P \rightarrow Q)}{0P \mid 1Q}$$

Beispiel:

$$\models (((\neg A \rightarrow B) \rightarrow C) \rightarrow ((\neg B \rightarrow A) \rightarrow C))$$

es soll also gezeigt werden, dass die Formel eine Tautologie ist. Wir nehmen an, das ist nicht der Fall; d.h. sie kann bei geeigneter Interpretation falsch werden. Dies deuten wir durch Voranstellen der 0 an:



alle Pfade führen zu Widersprüchen $\Rightarrow$ ursprüngl. (nicht negierte) Formel ist Tautologie.

Determinismus von Kalkül und Regeln:

Determinismus

- Die Regeln sind alle deterministisch
- Der Kalkül aber nicht:
  Wahl der nächsten Formel, auf die Regel angewendet wird

Heuristik

Nicht-verzweigende Regeln zuerst: „α vor β“

Beachte: dieselbe Formel kann mehrfach (auf verschiedenen Ästen) verwendet werden.

Formale Definition des Kalküls:

Tableau: binärer Wurzelbaum, dessen Knoten mit Formeln markiert sind.

Tableau-Ast: maximaler gerichteter Pfad in einem Tableau.

Sei M eine Formelmenge, sei A eine Formel

Initialisierung

Das Tableau, das nur aus dem Knoten $0A$ besteht, ist ein Tableau für A über M (d.h., für $M \models A$)

Erweiterung

- T ein Tableau für A über M
- B ein Ast von T
- F eine Formel auf B, die kein Literal ist

T' entstehe durch Erweiterung von B gemäß der auf F anwendbaren Regel (α oder β)
Dann ist T' ein Tableau für A über M

Voraussetzungsregel

- T ein Tableau für A über M
- F eine Formel in M

T' entstehe durch Erweiterung eines beliebigen Astes durch $1F$
Dann ist T' ein Tableau für A über M

Definition: Geschlossener Ast

Ast B eines Tableaus ist geschlossen, wenn

$$1F, 0F \in B \quad \text{oder} \quad \mathbf{10} \in B \quad \text{oder} \quad \mathbf{01} \in B$$

Definition: Geschlossenes Tableau

Ein Tableau ist geschlossen, wenn jeder seiner Äste geschlossen ist

Definition: Tableaubeweis

Ein Tableau für A über M, das geschlossen ist,
ist ein Tableaubeweis für $M \cup \{\neg A\} \vdash_{T0} \mathbf{0}$ und damit für $M \models A$

Satz (Korrektheit und Vollständigkeit des Tableaukalküls):

Es gilt $M \models A$
genau dann, wenn
es einen Tableaubeweis für A über M gibt

Kern des Korrektheitsbeweises:

Definition: Erfüllbares Tableau

Tableauast ist erfüllbar, wenn die Menge seiner Formeln erfüllbar ist
Tableau ist erfüllbar, wenn es (mindestens) einen erfüllbaren Ast hat

Lemma

Jedes Tableau für A über M ist erfüllbar,
falls $M \cup \{\neg A\}$ erfüllbar ist.

Lemma

Ein geschlossenes Tableau ist nicht erfüllbar

Kern des Vollständigkeitsbeweises:

Definition: Voll expandiertes Tableau

Ein Tableau heißt voll expandiert, wenn

- jede Regel
- auf jede passende Formel
- auf jedem offenen Ast

angewendet worden ist und

- für jedes $F \in M$ (hierfür muss M endlich sein)
- für jeden Ast B

$1F$ auf B vorkommt

Lemma

B ein offener Ast in einem voll expandiertem Tableau, dann ist B erfüllbar

Also

Ist $M \cup \{\neg A\}$ unerfüllbar
und also jeder Ast eines jeden Tableaus für A über M unerfüllbar,
dann ist jedes voll expandierte Tableau für A über M geschlossen
(denn sonst wäre er wegen des Lemmas erfüllbar)

Beweis des Lemmas:

Sei B ein offener Ast eines voll expandierten Tableaus

Wir definieren

$$I(P) := \begin{cases} W & \text{falls } 1P \in B \\ F & \text{falls } 0P \in B \\ \text{bel.} & \text{sonst} \end{cases}$$

Durch Induktion zeigt man leicht:
$val_I(F) = W$ für jedes F auf B.

Es folgt, dass I Modell von $M \cup \{\neg A\}$ ist.

Sequenzenkalkül

- Sequenzenkalküle spielen eine wichtige Rolle in der Beweistheorie
- gehen zurück auf G. Gentzen 1935
- haben nichts mit "Folgen" zu tun
  (engl.: *sequent*, nicht *sequence*)

Definition:

Eine *Sequenz* ist ein Paar endlicher Mengen von Formeln und wird notiert in der Form

$$\Gamma \Rightarrow \Delta.$$

Γ wird Antezedent und Δ Sukzedent genannt.

Sowohl links wie rechts vom Sequenzenpfeil $\Rightarrow$ kann auch die leere Menge stehen. Wir schreiben dann

$$\Rightarrow \Delta \quad \text{bzw.} \quad \Gamma \Rightarrow \quad \text{bzw.} \quad \Rightarrow .$$

Ist $\Gamma = \{A_1, \ldots, A_n\}$ und B eine Formel, so schreiben wir

$$\Gamma, B \quad \text{für die Menge} \quad \{A_1, \ldots, A_n, B\},$$

Entsprechend werden

$$B, \Gamma \qquad \Gamma, \Delta \qquad \Gamma, B, \Delta$$

benutzt.

Semantik von Sequenzen:

Sei I eine Interpretation.

Definition:Auswertung von Sequenzen

$$
\begin{array}{ll}
val_I(\Gamma \Rightarrow \Delta) & = val_I(\bigwedge \Gamma \rightarrow \bigvee \Delta) \quad \text{für } \Gamma, \Delta \neq \emptyset \\
val_I(\Gamma \Rightarrow) & = val_I(\neg \bigwedge \Gamma) \\
val_I(\Rightarrow \Delta) & = val_I(\bigvee \Delta) \\
val_I(\Rightarrow) & = \mathbf{F}
\end{array}
$$

$$
\begin{array}{lcl}
\bigwedge \Gamma & = & \text{Konjunktion aller Formeln in } \Gamma \\
\bigvee \Gamma & = & \text{Disjunktion aller Formeln in } \Gamma
\end{array}
$$

Die Konjunktion über die leere Folge wird demnach zu **W** und die Disjunktion über die leere Folge zu **F** ausgewertet.

Axiome und Regeln des Kalküls:

Axiome:

$$\Gamma, A, \Gamma' \Rightarrow \Delta, A, \Delta'.$$

Die Regeln sind

$$(\neg links) \quad \frac{\Gamma, \Gamma' \Rightarrow A, \Delta}{\Gamma, \neg A, \Gamma' \Rightarrow \Delta}$$

$$(\wedge links) \quad \frac{\Gamma, A, B, \Gamma' \Rightarrow \Delta}{\Gamma, (A \wedge B), \Gamma' \Rightarrow \Delta}$$

$$(\vee links) \quad \frac{\Gamma, A, \Gamma' \Rightarrow \Delta \quad \Gamma, B, \Gamma' \Rightarrow \Delta}{\Gamma, (A \vee B), \Gamma' \Rightarrow \Delta}$$

$$(\rightarrow links) \quad \frac{\Gamma, \Gamma' \Rightarrow A, \Delta \quad B, \Gamma, \Gamma' \Rightarrow \Delta}{\Gamma, (A \rightarrow B), \Gamma' \Rightarrow \Delta}$$

(Fortsetzung:)

$$(\neg rechts)\ \frac{A, \Gamma \Rightarrow \Delta, \Delta'}{\Gamma \Rightarrow \Delta, \neg A, \Delta'}$$

$$(\wedge rechts)\ \frac{\Gamma \Rightarrow \Delta, A, \Delta' \quad \Gamma \Rightarrow \Delta, B, \Delta'}{\Gamma \Rightarrow \Delta, (A \wedge B), \Delta'}$$

$$(\vee rechts)\ \frac{\Gamma \Rightarrow \Delta, A, B, \Delta'}{\Gamma \Rightarrow \Delta, (A \vee B), \Delta'}$$

$$(\rightarrow rechts)\ \frac{A, \Gamma \Rightarrow B, \Delta, \Delta'}{\Gamma \Rightarrow \Delta, (A \rightarrow B), \Delta'}$$

Zusätzliche Regeln für eine Menge M von Voraussetzungen

$$(\text{einfügen}_M)\ \frac{B, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}$$

für $B \in M$

Def. Ableitbarkeit:

Für $A \in For0$, $M \subseteq For0$ sagen wir, dass A *aus* M *ableitbar in* **S**0 ist,

$$M \vdash_S A,$$

gdw.

eine Ableitung von $\Rightarrow A$ aus M in **S**0 existiert.

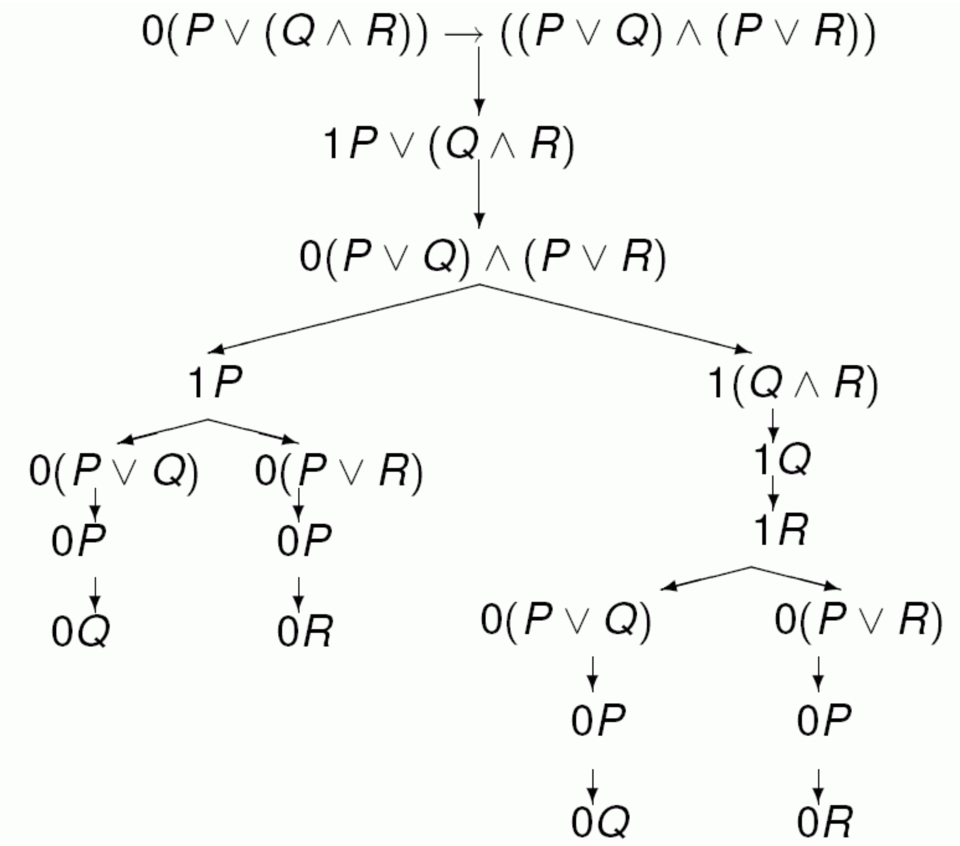
Satz (Korrektheit und Vollständigkeit):

S0 ist korrekt und vollständig:
es gilt also

$$M \models A \quad \Leftrightarrow \quad M \vdash_{\mathbf{S0}} A.$$

Transformation vom Tableau-Kalkül in den Sequenzenkalkül:

Beisp. eines Tableau-Beweises



Seq.-Ableitung von $(P \vee (Q \wedge R)) \rightarrow ((P \vee Q) \wedge (P \vee R))$:

(1)	$P \Rightarrow P, Q$	(Axiom)
(2)	$P \Rightarrow P, R$	(Axiom)
(3)	$Q, R \Rightarrow P, Q$	(Axiom)
(4)	$Q, R \Rightarrow P, R$	(Axiom)
(5)	$P \Rightarrow P \vee Q$	($\vee$ re 1)
(6)	$P \Rightarrow P \vee R$	($\vee$ re 2)
(7)	$Q, R \Rightarrow P \vee Q$	($\vee$ re 3)
(8)	$Q, R \Rightarrow P \vee R$	($\vee$ re 4)
(9)	$P \Rightarrow ((P \vee Q) \wedge (P \vee R))$	($\wedge$ re 5,6)
(10)	$Q \wedge R \Rightarrow ((P \vee Q) \wedge (P \vee R))$	($\wedge$ re 7,8)
(11)	$(P \vee (Q \wedge R)) \Rightarrow ((P \vee Q) \wedge (P \vee R))$	($\vee$ li 9,10)
(12)	$\emptyset \Rightarrow (P \vee (Q \wedge R)) \rightarrow ((P \vee Q) \wedge (P \vee R))$	($\rightarrow$ re 11)

Beweis in Baumdarstellung:

$$\emptyset \Rightarrow (P \vee (Q \wedge R)) \rightarrow ((P \vee Q) \wedge (P \vee R))$$

$$(P \vee (Q \wedge R)) \Rightarrow ((P \vee Q) \wedge (P \vee R))$$

$$P \Rightarrow ((P \vee Q) \wedge (P \vee R)) \qquad Q \wedge R \Rightarrow ((P \vee Q) \wedge (P \vee R))$$

$$P \Rightarrow P \vee Q \quad P \Rightarrow P \vee R$$

$$P \Rightarrow P, Q \quad P \Rightarrow P, R \qquad Q, R \Rightarrow (P \vee Q) \wedge (P \vee R)$$

Axiom Axiom

$$Q, R \Rightarrow P \vee Q \quad Q, R \Rightarrow P \vee R$$

$$Q, R \Rightarrow P, Q \quad Q, R \Rightarrow P, R$$

Axiom *Axiom*

Transformation von Tableau zu Sequenz:

T ein beliebiges Tableau. Wir definieren einen Ableitungsbaum $Seq(T)$ im Sequenzenkalkül wie folgt:
Bei der Anwendung einer α-Regel beim Aufbau des Tableaus T, werden jeweils zwei Knoten hinzugefügt, der erste und der zweite α-Knoten. Die Knoten von $Seq(T)$ sind alle Knoten von T mit Ausnahme der ersten α Knoten.
Ein Knoten, N, in $Seq(T)$ wird mit der Sequenz

$$A_1, \ldots, A_k \Rightarrow B_1, \ldots B_n$$

markiert, wobei $A_1, \ldots, A_k$ alle Formeln sind, so dass $1A_i$ auf dem Teilpfad P vorkommt, der von N zur Wurzel von T führt und $B_1, \ldots B_n$ alle Formeln sind, so dass $0B_i$ auf P liegt und noch keine Tableauregel auf $1A_i$ und $0B_i$ angewendet wurde.

Theorem (Korrektheit der Transformation Seq)

Ist T ein abgeschlossenes Tableau, so ist $Seq(T)$ ein Beweisbaum im Sequenzenkalkül.

Ausschnitte entnommen aus
Beckert (2010), Görz (2010), Otto (2010) und Schmitt (2008)
(genaue Quellenangaben siehe http://www.uni-forst.gwdg.de/~wkurth/fs10_lit.htm)