Optimierungen: Links zwischen Formeln

Definition. Sei T ein Tableau. Ein Paar Formeln $\langle \phi, \psi \rangle$ auf demselben Ast $B \in T$ ist ein **Link**, wenn es Unterformeln ϕ', ψ' von ϕ, ψ mit entgegengesetzter Polarität gibt, sowie eine Substitution σ, die $\langle \phi', \psi' \rangle$ zu einem komplementären Paar macht. ■

`select_formula` kann Formeln ignorieren, die kein Link zu einer Formel auf dem aktuellen Ast besitzen.

Beispiel:

$$B = \{\forall x \underbrace{\overbrace{p(f(x))}}, \overbrace{\neg p(y)}, \underbrace{\neg p(f(z))}, q(a) \wedge \neg p(g(u))\}$$

$q(a) \wedge \neg p(g(u))$ hat keinen Link und kann ignoriert werden.

Optimierungen: Universelle Formeln

Definition. Sei T ein FV-Tableau und ϕ eine Formel auf einem Ast B von T.
Konstruiere T' durch hinzufügen von $\forall x \phi$ zu B, wobei $x \in$ Var.

Dann heißt ϕ **universell** auf B bzgl. x, falls $T \models T'$ ■

Intuition:
ϕ könnte auf B mit x universell quantifiziert werden.

Proposition. Eine Formel ϕ auf einem Ast B ist universell bzgl. x,
falls für alle Grundsubstitutionen σ gilt:
$B\sigma \models (\forall x \phi)\sigma$. ■

Mgus dürfen bei Anwendung der Abschlußregel auf nicht-universelle Variablen beschränkt werden.

$$\forall x p(x)$$

$$\neg p(a) \vee \neg p(b)$$

$$p(X)$$

$$\neg p(a) \qquad \neg p(b)$$

$$\epsilon \qquad\qquad \epsilon$$

jeweils $p(X)$ universell bzgl. X

Universalität ist i.a. unentscheidbar, jedoch:

Hinreichendes Kriterium:
$\phi \in B$ ist universell bzgl. x, falls ϕ entstanden ist durch Anwendung

- einer γ-Regel, wobei x die neu eingeführte freie Variable ist,
- einer γ-, $\delta-$ oder α-Regel auf eine Formel, die auf B bzgl. x universell ist, oder
- einer β-Regel auf eine Formel β, die auf B bzgl. x universell ist, falls x in $\beta_i \neq \phi$ nicht auftritt.

Tableauprozedur für SNF

Zur Erinnerung:

Klauselmengen sind abgeschlossene Formeln in SNF.

Mengenschreibweise:

$$\{\{p(x), \neg q(y)\}, \{\neg p(x)\}\}$$

anstatt

$$(\forall x \forall y (p(x) \vee \neg q(y))) \wedge (\forall z \neg p(z))$$

Konvention:

Für Klausel

$$C = \forall x_1, \ldots, x_m (l_1 \vee \cdots \vee l_r)$$

ist σ-Instanz

$$C\sigma = \{l_1\sigma, \ldots, l_r\sigma\}$$

Vorteile von SNF

- Einfachere
  - Tableauprozedur
  - Korrektheits- und Vollständigkeitsbeweise
  - Aufdeckung und Beschreibung von Verstärkungen und Verfeinerungen
- Effizientere Implementierung (Prolog-Technologie)

Probleme von SNF

- Verständlichkeit der Tableaubeweise
- Nichtklassische Logiken

Definition. Ein **Klauseltableau** für eine endliche Klauselmenge $S = \{C_1, \ldots, C_m\}$ ist ein endlichverzweigender, mit Literalen markierter Baum, konstruiert mittels:

1. Baum mit genau einem Knoten, markiert mit $\top$, ist Klauseltableau für S.

2. T Klauseltableau für S, B Zweig von T, $C = \{l_1, \ldots, l_n\} \in S$, ν Variablenumbenennung. Man erhält neues Klauseltableau für S, indem B durch n neue Nachfolger erweitert wird mit Markierungen $l_1\nu, \ldots, l_n\nu$.

$$\frac{\overbrace{\forall x_1, \ldots, x_m (l_1 \vee \cdots \vee l_r)}^{C \in S}}{l_1\nu \;\Big|\; \cdots \;\Big|\; l_r\nu} \quad \nu \text{ Variablenumben.}$$

3. (Substitutionsregel) Sei T Klauseltableau für S, σ Substitution. Dann ist $T\sigma$ Klauseltableau für S.

■

Eingabeklauseln aus S nicht Teil des Tableaus.

Alle Knoten mit Literalen markiert.

2. aus $\beta + \gamma +$(Substitutionsregel).

Abgeschlossene Zweige enthalten komplementäres Literalpaar.

```
type clause;  /* universally quantified */
              /* disjunction of literals */
type instance = list of literal;
type input =  list of clause;
type branch = list of literal;
type tableau = list of branch;

var P,Q: literal;
var S: input;
var B: branch;
var T: tableau;
var σ: substitution;

function extend(B:branch,S:input):tableau;
var U: tableau;
var C: clause;
var I: instance;
begin
  C := select_clause(B,S);
  I := new_instance_of(C);
  U := [];
  while I≠[] do begin
   U := append(U,append(B,[first(I)]));
   I := rest(I)
  end;
  return(U)
end;
```

select_clause ersetzt select_formula

```
/* initialize */

S := [];
while ¬eof do
  S := append(S,[read_next_clause]);

/* clausal tableaux */

T := [[]];
while T≠[] do begin
  B := select_branch(T);
  T := delete(B,T);
  if select_mode(B) = close then begin
    [P,Q] := select_pair(B);
    σ := mgu(P,Q);
    T := apply_subst(σ,T)
    end
  else
    T := append(T,extend(B,S))
end
writeln('Proof.')
```

Satz. (Lifting)
Sei T' ein beliebiges geschlossenes Klauseltableau.

Die Klauseltableauprozedur kann

– für eine geeignete Substitution σ,

– bei geeigneter Implementierung von `select_clause`, `select_mode` und `select_pair` oder mit Hilfe von Backtracking

ein geschlossenen Klauseltableau T erzeugen, so daß

$$T' = T\sigma$$

■

Lemma. Sei T ein Tableau und τ eine Substitution, so daß $T\tau$ geschlossen ist.
Der Ast $B\tau$ von $T\tau$ sei durch den Link $\langle L\tau, L'\tau\rangle$ geschlossen. ρ sei Mgu von $\langle L, L'\rangle$.

Dann gilt $T\rho\tau = T\tau$. ■

Beweis: (Lemma) ρ ist ein Mgu, also existiert τ' mit $\tau' \circ \rho = \tau$. O.b.d.A. ist ρ idempotent, daher
$T\rho\tau = T\rho\rho\tau' = T\rho\tau' = T\tau$. ■

Beweis: (Lifting-Satz)
Sei T'' wie T' aber ohne Substitutionen; und sei τ die Substitution, so daß $T''\tau = T'$.

`select_clause`, `select_mode` seien so gewählt, daß Zweig B_1 von T'' erzeugt wird.

Da T' geschlossen ist,
enthält $B_1\tau$ komplementäres Paar $\langle L_1\tau, L'_1\tau\rangle$.

Sei ρ_1 Mgu von $\langle L_1, L'_1\rangle$,
welches zudem Ergebnis von `select_pair` sei.

Sei $T_1 = T''\rho_1$ erzeugt durch Anwendung der Abschlußregel.

Aus dem Lemma folgt: $T_1\tau = T''\rho_1\tau = T''\tau = T'$.

Induktion über die Äste von T'' liefert ein $T = T_n$,
das von der Prozedur gefunden wird,
so daß $T\tau = T'$ und T geschlossen. ■

Satz. (Vollständigkeit)
Sei S eine unerfüllbare Klauselmenge.

Dann findet die Prozedur ein geschlossenes Klauseltableau T für S. ■

Beweis: Sei Σ die Signatur von S.

Aus Satz v. Herbrand folgt:
Es gibt eine endliche unerfüllbare Grundklauselmenge $S' \subset \mathsf{Form}^0(S)$.

S' ist *aussagenlogisch unerfüllbar* über der Signatur $\Sigma' = \{p \mid p \text{ Atom in } S'\}$.

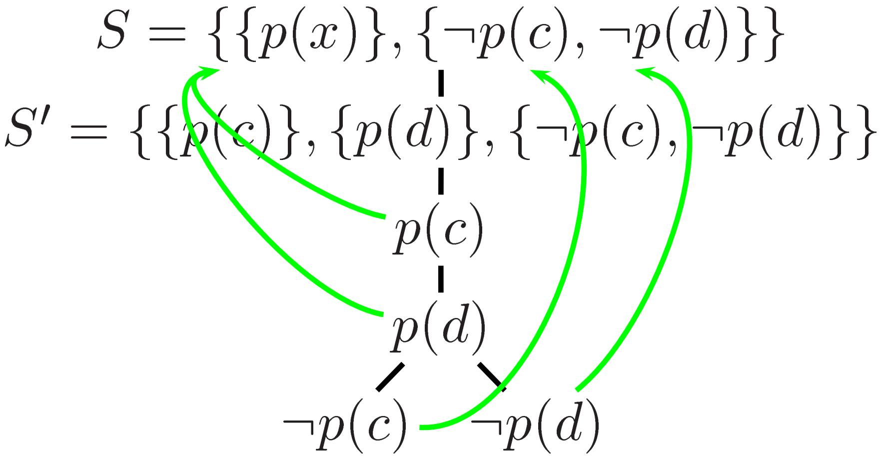
Aus Vollständigkeit *aussagenlogischer* Klauseltableaus: es gibt geschlossenes Klauseltableau T' für S'.

Für alle Klauseln $C^{i\prime}$, die in T' verwendet sind, sei τ^i eine Substitution, so daß $C^i\tau^i = C^{i\prime}$ für ein $C^i \in S$.
Und es sei $\tau = \bigcup_{C^{i\prime} \text{ verwendet in } T'} \tau^i$.

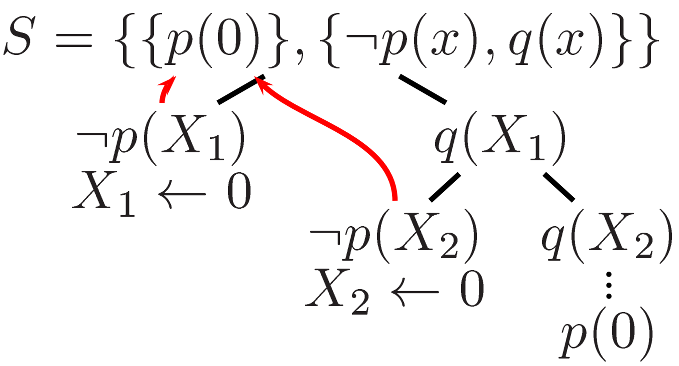
T'' sei wie T' aber ohne Substitutionen.
Da $T''\tau = T'$,
ist T'' ein geschlossenes Klauseltableau für S.

Aus Lifting-Satz:
Die Klauseltableauprozedur findet geschlossenes Tableau T für S. ■

Beispiel.



Herbrandmodelle entsprechen Grundinstanzen unendlicher, fairer und offener Zweige:



$\langle\{0\}, I\rangle$ mit $I(p) = I(q) = \{0\}$ ist Modell von S.

Verfeinerungen von Klauseltableaus

Uniforme Gestalt universeller Klauseln ermöglicht einfache Definition von Verfeinerungen.

Was sind Verfeinerungen?

Einschränkung der Menge aller Klauseltableaus, so daß es immer noch für jede unerfüllbare Klauselmenge ein geschlossenes Tableau gibt.

Vollständigkeit bleibt erhalten, aber Anzahl zu betrachtender Tableaus nimmt ab, d.h. Suchraum wird kleiner.

Korrektheit bleibt erhalten; Lifting-Schema gleich.

Alternative Sichtweise:

Elimination von Choicepoints in

select_clause(S)

select_pair(B)

Konkretisierung von:

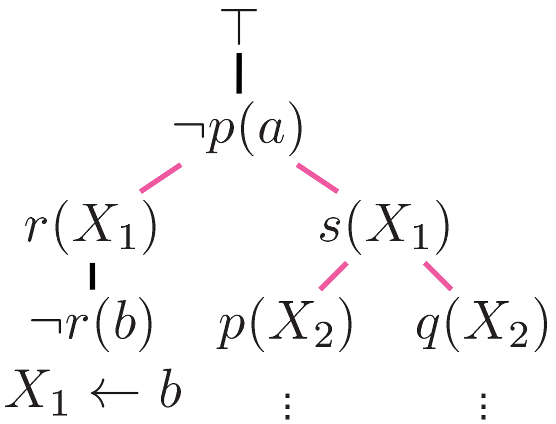
select_branch(T)

Konnektionstableaus

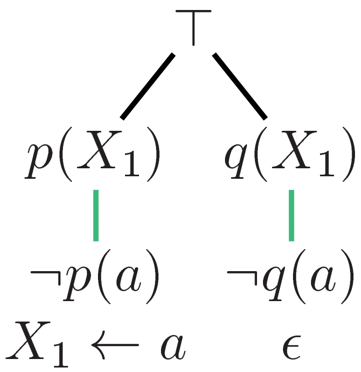
Definition. Ein **Konnektionstableau** ist ein Klauseltableau, worin jeder innere mit L markierte Knoten (außer $\top$) ein mit L' markiertes Blatt als unmittelbaren Nachfolger hat, so daß $\langle L, L' \rangle$ komplementäres Paar ist. ■

Beispiel:

$$S = \{\{p(x), q(x)\}, \{r(x), s(x)\},\\ \{\neg p(a)\}, \{\neg q(a)\}, \{\neg r(b)\}, \{\neg s(b)\}\}$$



Kein Konnektionstableau.



Konnektionstableau. Weniger Redundanz möglich.

```
function select_clause(B,S:branch):
  ⟨clause,substitution⟩;
var S': branch;
begin
  if B = [] then
    return(or(S×{ϵ}))
  else begin
    S' := {⟨C,σ⟩ | C ∈ S und
                 ⟨C,last(B)⟩ Link mit mgu σ};
    return(or(S'))
  end
end;

function extend(B,S:branch):
  ⟨tableau,substitution⟩;
    ...
  return(⟨U,σ⟩)
end;

/* connection tableaux */

T := first(extend([],S));
  ...
    else
      begin
      ⟨U,σ⟩ := extend(B,S);
      T := apply_subst(σ,append(T,U))
      end
  ...
```

Jeder Erweiterungsschritt ermöglicht einen Abschluß mit einem Blatt des erweiterten Tableaus.

(Nicht so bei Link-Optimierung im Nichtklauselfall.)

Abschlüsse, die *nicht* von dieser Art sind, heißen **Reduktion**

Satz. Konnektionstableaus sind nicht beweiskonfluent. ■

Beweis: Sei $S = \{\{p\}, \{\neg p\}, \{q\}\}$, und $\{q\}$ sei als erste Klausel selektiert. ■

Daher: Vollständige Implementierung von Konnektionstableaus benötigt Backtracking. Faire Implementierung von `select_clause` reicht nicht aus.

Offener, nicht erweiterbarer Zweig ergibt i.a. kein Modell. Daher: Konnektionstableaus ungeeignet zur Suche nach „Gegenbeispielen“ für nicht allgemeingültige Formelmengen.

Schwache Konnektionstableaus

Erweiterungsschritt nur mit Klauseln erlaubt, die einen Link in den aktuellen Ast haben.
Aber: mit *beliebigem* Ast-Literal
(nicht notwendigerweise mit dem Blatt).

Definition. Ein **schwaches Konnektionstableau** ist ein Klauseltableau, in dem jeder Knoten – außer der Wurzel und den Blättern – einen direkten Nachfolger L hat, so daß es ein Literal L' auf dem Pfad zwischen L und der Wurzel gibt, das mit L ein komplementäres Paar $\langle L, L' \rangle$ bildet. ■

Definition. Eine Klauselmenge heißt **minimal unerfüllbar**, wenn sie unerfüllbar ist und jede ihrer echten Teilmenge erfüllbar. ■

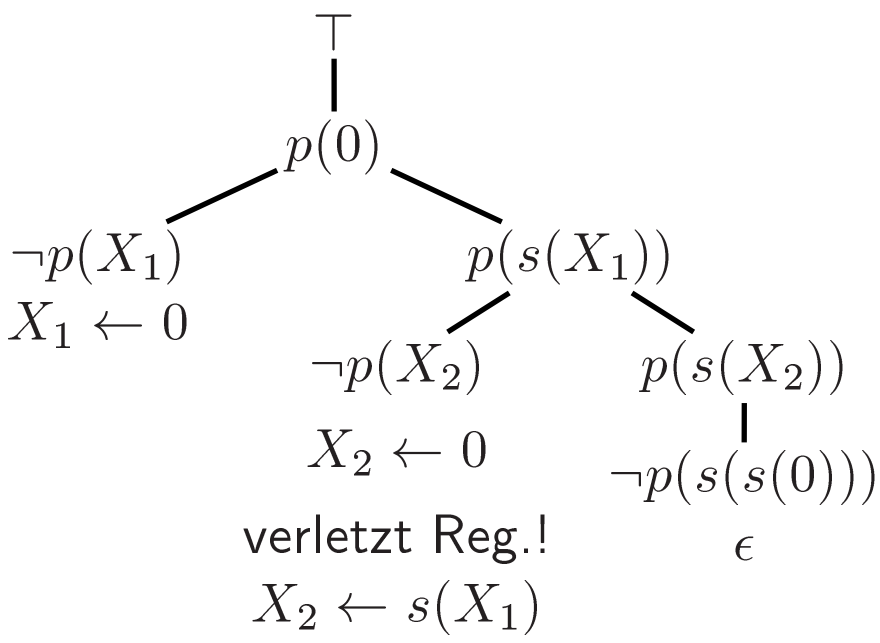
Proposition. Schwache Konnektionstableaus sind beweiskonfluent, falls die Start-Klausel aus einer minimal unerfüllbarer (Teil-)Menge stammt. ■

Regularität

Definition. Ein Klauseltableau heißt **regulär**, wenn auf keinem seiner Äste dasselbe Literal mehr als einmal vorkommt. ■

Beispiel.

$S = \{\{p(0)\}, \{\neg p(x), p(s(x))\}, \{\neg p(s(s(0)))\}\}$



Regularität verhindert redundante identische Instantiierung.