Proof Confluent Tableau Calculi
—A Tutorial—

Reiner Hähnle & Bernhard Beckert

Universität Karlsruhe

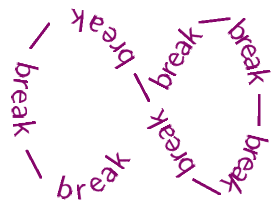
Institut für Logik, Komplexität und
Deduktionssysteme

reiner@ira.uka.de
beckert@ira.uka.de

TABLEAUX Conference

June 1999

Contents



Basics: Notation

$$\left.\begin{array}{l}\textbf{Terms } t \text{ — } c,\ f(x,g(a,y)),\ldots\\ \textbf{Atoms } A \text{ — } p,\ q(f(x,g(a,y))),\ldots\\ \textbf{Literals } L \text{ — } p,\ \neg q(f(x,g(a,y))),\ldots\\ \textbf{ground} \text{ term = variable-free term, etc.}\\ \textbf{complement} \text{ of literal: } \overline{L}\\ \textbf{Substitution } \sigma,\ \tau\end{array}\right\}\text{ as usual}$$

Notation for substitutions:

$$\sigma(x) = \begin{Bmatrix} t_i & x = x_i,\ 1 \le i \le n \\ x & \text{otherwise} \end{Bmatrix} = \{x_1/t_1, \ldots, x_n/t_n\}$$

Renaming: for each x, $\sigma(x)$ is “new” variable

σ is **unifier** of set of terms T if $|T\sigma| = 1$

σ is **most general unifier (MGU)** of T if:

1. σ is unifier of T
2. for all unifiers τ of T, there is ρ s.th. $\tau = \rho \circ \sigma$

Basics: Notation/Semantics

Clause:
$(\forall x_1) \cdots (\forall x_n) L_1 \vee \cdots \vee L_r$,
where L_i literals and Var$(\{L_1, \ldots . L_r\}) = \{x_1, \ldots , x_n\}$.

$L \in C$ means: L occurs in C

New instance of a clause:
$(L_1 \vee \cdots \vee L_r)\sigma$, where σ is a renaming.

If σ ground substitution: **ground instance**.

(Partial) Herbrand interpretation $\mathbf{I}$:
consistent set of ground literals

$\mathbf{I} \models L$ iff $L \in \mathbf{I}$

$\mathbf{I} \models C$ (C ground) iff $L \in \mathbf{I}$ for some $L \in C$

$\mathbf{I} \models C$ iff $\mathbf{I} \models C'$ f.a. ground instances C' of C

$\mathbf{I} \models S$ (S set of clauses) iff $\mathbf{I} \models C$ f.a. $C \in S$

S is **satisfiable** (by $\mathbf{I}$)

Basics:
Free Variable Clause Tableaux

(Clause) tableau for set of clauses S built by a (possibly infinite) sequence of applications of construction rules:

Initialization: Tree with single node $\top$ is tableau for S

Expansion: T tableau for S, B branch of T, and $L_1 \vee \cdots \vee L_r$ new instance of clause in S.

T' is a tableau for S if obtained by extending B with r nodes $L_1, \ldots, L_r$.

Closure: T tableau for S, B branch of T, and L, L' literals on B.

If $\{L, \overline{L'}\}$ has MGU σ, then T' is a tableau for S if obtained by appending $\perp$ to B and applying σ to each node of T.

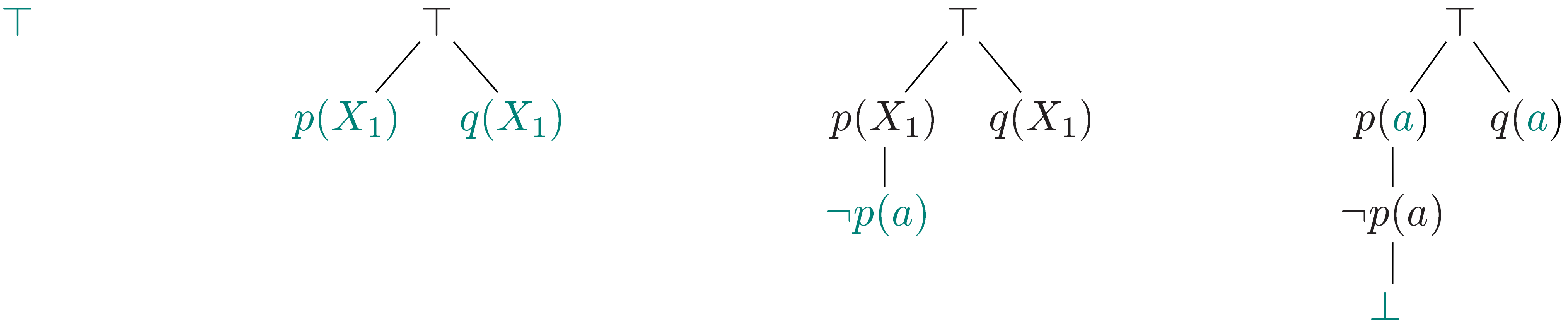
B **closed** if it contains $\perp$.

T **closed** if all branches of T are closed.

T **proof** (for unsat. of S) if T is closed tableau for S.

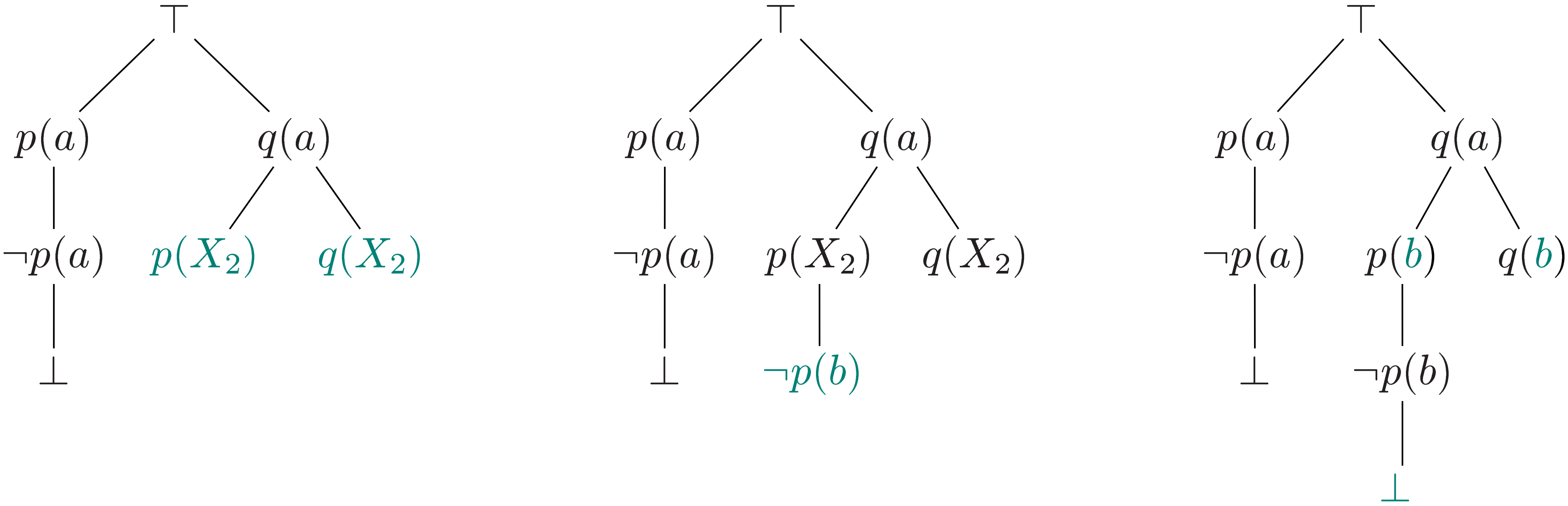
Example

$$S = \{p(x) \lor q(x), \quad \neg p(a), \quad \neg p(b), \quad \neg q(b)\}$$



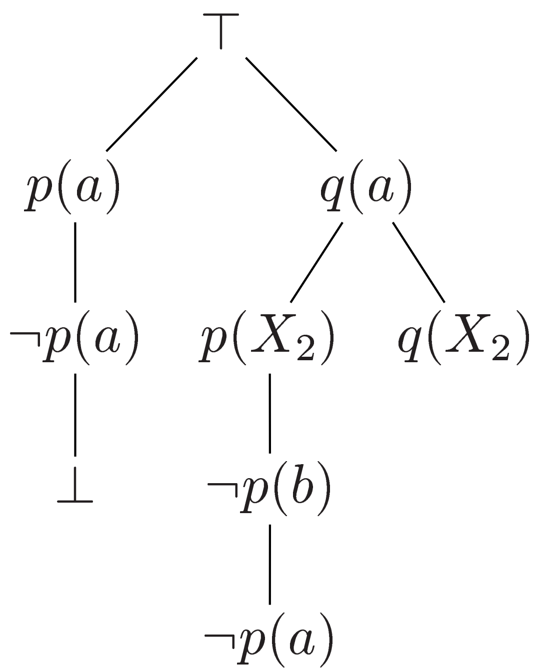
Example

$$S = \{p(x) \vee q(x), \quad \neg p(a), \quad \neg p(b), \quad \neg q(b)\}$$



Properties of Tableau Calculi: Sources of Indeterminism

- choose branch on which to apply next rule
- choose mode — expansion/closure
- choose clause for expansion
- choose MGU of literals for closure



Variants of Basic Clause Tableaux

Ground or **propositional** tableaux:

Input clause set S is ground.

Consequence: $\sigma = \{\}$ the only MGU ever needed.

Smullyan or **sentence** tableaux:

Ground instances used in expansion rule

Consequence: as above.

Delayed closure tableau [Fitting, 1996]:

No closure rule.

A tableau T is **closed** if there is a substitution σ s.th. all branches of $T\sigma$ contain literals L and $\overline{L}$.

Properties of (Tableau) Calculi

A calculus is **complete** if every unsatisfiable clause set S has a proof.

Analysis of indeterminism yields proof enumeration procedures.

A calculus is **proof confluent** if every tableau for an unsatisfiable clause set S can be completed to a proof.

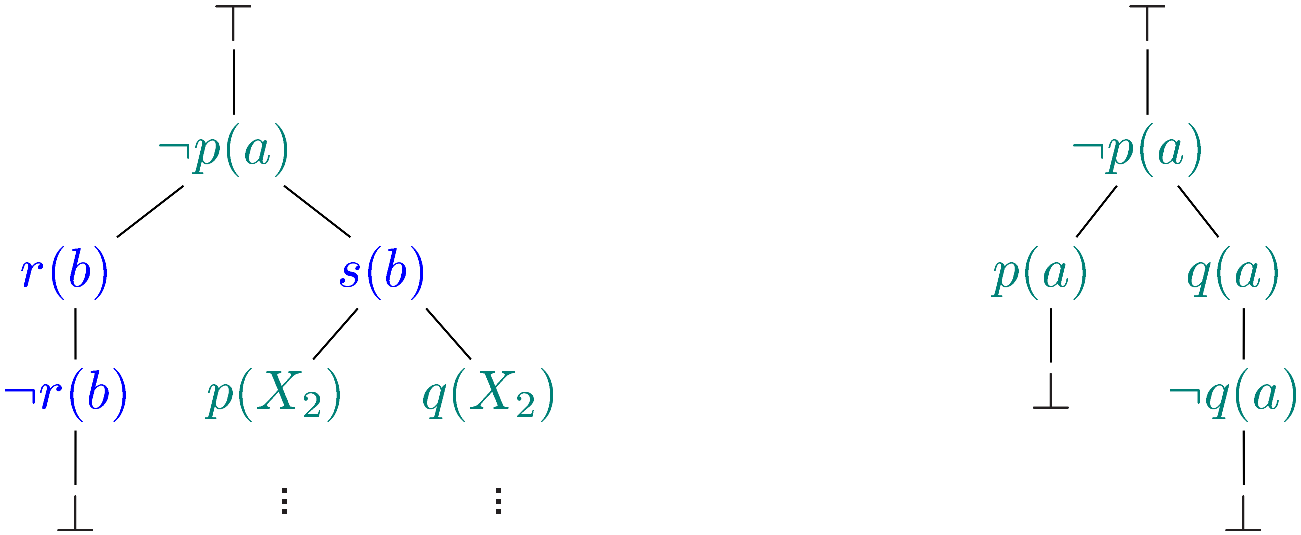
A calculus is **non-destructive** if all tableaux that can be constructed from a given T contain T as an initial subtree; otherwise it is **destructive**.

Free variable clause tableaux are complete,
proof confluent, and destructive.

Ground tableaux, Smullyan (sentence) tableaux, and tableaux w/delayed closure are complete, proof confluent, and *non-destructive*.

Example & Motivation for Refinements

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



Basic Refinements: Connection Tableaux

Definition.
A **connection tableau** is a clause tableau, in which every inner node L (except $\top$) has $\overline{L}$ as one of its immediate successors. ■

Procedural definition:

Extension:
$T \neq \top$ connection tableau for S, B branch of T ending with L , and $L_1 \vee \cdots \vee L_r$ new instance of clause in S, σ MGU of $\overline{L}$ and some L_i.

$T'\sigma$ is a connection tableau for S, where T' obtained by extending B with r nodes $L_1, \ldots, L_r$ and the branch ending in L_i with $\perp$.

Reduction: The new name for Closure.

Weak Extension:
$T \neq \top$ weak connection tableau for S, B branch of T containing L, and $L_1 \vee \cdots \vee L_r$ new instance of clause in S, σ MGU of $\overline{L}$ and some L_i.
[. . .]

Ground (weak) connection tableaux not proof confluent

Consider $S = \{p, \neg p, q\}$ with q used in first extension.

Why bother with proof confluent calculi?

1. They form basis for (incomplete) search for models of satisfiable problems (counter examples).

Open branches, on which rules were exhaustively applied, indicate satisfiability of input.

2. Proof confluence is a necessary prerequisite for strong completeness.

A **proof procedure** is a "deterministic calculus":

There is a function F that, given a clause set S and a tableau T, computes in poly-time (in the size of S and T) the next rule to be applied on T.

A complete proof procedure is called **strongly complete**.

Strongly complete proof procedures avoid enumeration of proofs.

Complete, proof confluent, and non-destructive calculi can usually be turned into strongly complete proof procedures.

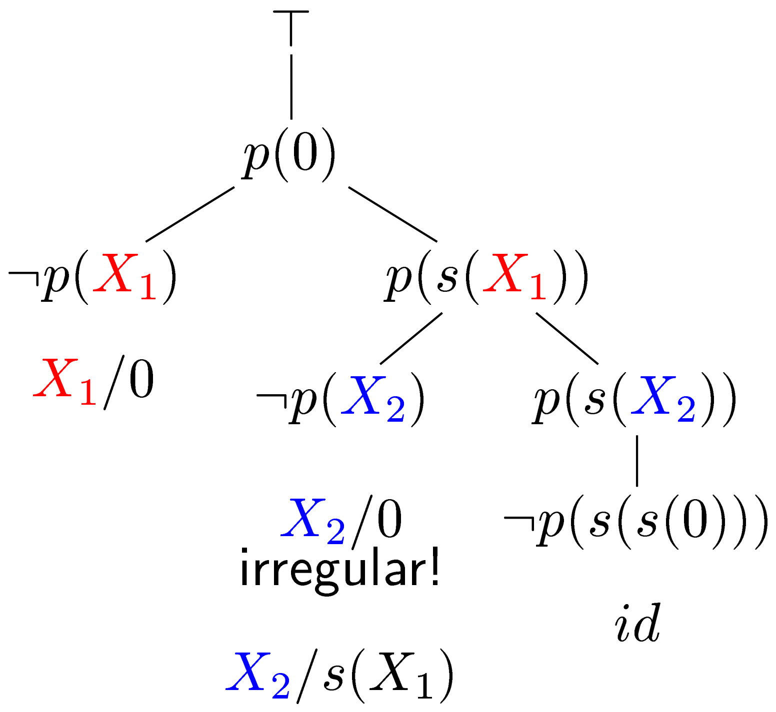
Is there a complete, proof confluent, and destructive tableau proof procedure?

Basic Refinements: Regularity

Definition.
A clause tableau is **regular** when all of its branches contain at most one occurrence of the same literal. ■

Example

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



Avoiding Redundancy

Regularity instance of general theme: avoid redundancy

Open branches B in ground clause tableau for S define partial interpretation $\mathbf{I}_B$:

$$\mathbf{I}_B \models L \text{ iff } L \in B$$

Regular tableaux + exclusion of tautologies:

avoid extension with C, if $\mathbf{I}_B \models C$.

Definition.
An open clause tableau branch B has a **saturation** wrt S iff there is an open $\widetilde{B} \supseteq B$ such that $\mathbf{I}_{\widetilde{B}} \models S$. ■

S satisfiable ⇒ initial branch $\top$ has saturation

Try only small number of candidates $\widetilde{B}$

Rationale:
$\widetilde{B} = B \cup$ selected literals from clauses C with $\mathbf{I}_B \not\models C$

Tableaux with Selection Function

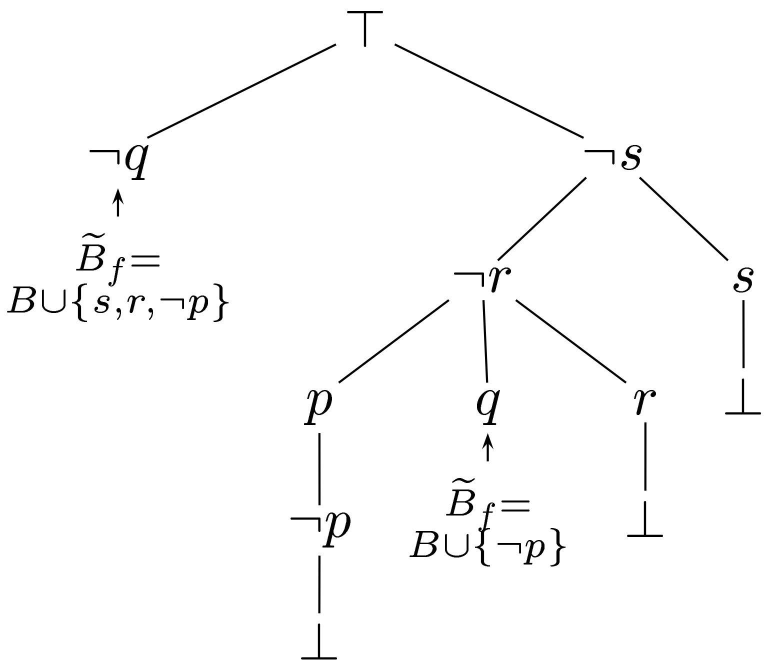
Definition.
A **selection function** f maps each clause into a (possibly empty) subset of its body literals.

The **saturation** of a branch B **wrt** f is

$$\widetilde{B}_f = B \cup \{f(C) \mid C \in S,\, \mathbf{I}_B \not\models C\}$$ ■

Example

$$S = \{\neg q \lor \neg s, \quad \neg r \lor s, \quad p \lor q \lor r, \quad \neg p\}$$



Selection Functions II

When $\widetilde{B}_f$ is consistent and all clauses C with $f(C) = \emptyset$ were used on B, then $\widetilde{B}_f \models S$.

Otherwise — Which clauses to use for extension?

Use clauses $C \in S$ with $\mathbf{I}_B \not\models C$ and one of:

1. no literals selected in C: $f(C) = \emptyset$
2. $f(C)$ weakly connected to $\widetilde{B}_f$: $\overline{f(C)} \cap \widetilde{B}_f \neq \emptyset$

   Restart step w/ **restart clause** C

   (a) $\overline{f(C)} \cap f(D) \neq \emptyset$ for $D \in S$

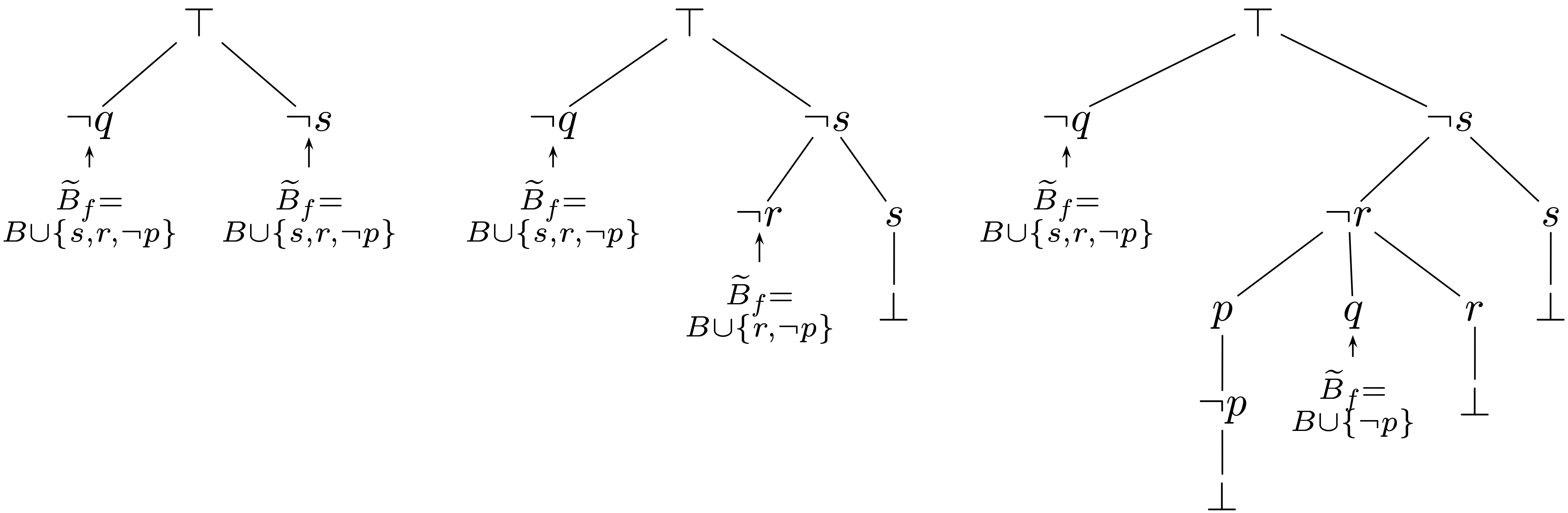
   (b) $\overline{f(C)} \cap B \neq \emptyset$ or

   Weak extension step

The first extension step in a tableau is always a restart step.

Example

$$S = \{\overbrace{\neg q \vee \color{red}{\neg s}, \quad \neg r \vee \color{red}{s}}^{\text{Restart clauses}}, \quad p \vee q \vee \color{red}{r}, \quad \color{red}{\neg p}\}$$



Completeness of Tableaux with Selection Function

Theorem. If the finite ground clause set S is unsatisfiable, then for any selection function f there is a tableau proof with selection function f for S. ■

Proof:
Assume to contrary there is a tableau w/selection function f, open branch B, no more rule applicable.

S is finite $\rightsquigarrow$ B is finite.

Sufficient to prove $\widetilde{B}_f$ is consistent.

Assume to contrary $\widetilde{B}_f$ were inconsistent.

There is literal $L \in \{f(C) \mid C \in S, \mathbf{I}_B \not\models C\}$ s. th.

(a) $\overline{L} \in B$ or

(b) $\overline{L} \in f(D)$ for some D

In either case, extension step possible, ↯ no more rule applicable on B ■

Proof independent of sequence of extension steps
$\rightsquigarrow$ tableaux with selection function proof confluent!

Ordered Tableaux

(Hähnle & Klingenbeck, 1996)

Motivated by ordered resolution.

Definition.
A ground **atom (A-)ordering** is a binary relation $<$ on atoms which is irreflexive and transitive. ■

Special case of tableaux with selection function:

- Ordering $<$ induces selection function

  $f_<(C) = \{L \mid \text{ atom of } L \text{ maximal in } C \text{ wrt } <\}$

- (Hähnle & Klingenbeck, 1996) do not require $\mathbf{I}_B \not\models C$

By def., only type 2. restart clauses ($f(C) \neq \emptyset$).

Can one get rid of restart steps altogether?

There is a total A-ordering such that w/o restart steps the calculus is incomplete.

Lifting

$$
\begin{tikzcd}
\text{Unsatisfiable finite clause set } S \arrow[r, dashed] \arrow[d, "\text{Herbrand's Theorem}"'] & \text{Tableau proof of } S \\
\text{Unsat. finite set of ground instances } \overline{S} \arrow[r, "\text{Ground Completeness}"] & \text{Tableau proof of } \overline{S} \arrow[u, "\text{Lifting}"']
\end{tikzcd}
$$

$$
\begin{tikzcd}
\rho\text{-instance of } S\text{: } L_1\rho \vee \cdots \vee L_i\rho \vee \cdots \vee L_r\rho \\
\text{Clause in } \overline{S}\text{: } L_1\sigma \vee \cdots \vee L_i\sigma \vee \cdots \vee L_r\sigma \arrow[u, "\text{Lifting}"']
\end{tikzcd}
$$

$f(C\sigma) \subseteq f(C)\sigma$, resp.,

$p < q$ implies $p\sigma < q\sigma$

$\sigma = \tau \circ \rho$ for some τ, $L_i\sigma$ selected/maximal

Selection Functions III

Definition.
A selection function is **consistent** (with respect to S) if, for each ground instance $\widehat{S}$ of S, the set $\bigcup_{C \in \widehat{S}} f(C)$, containing the selected literals of $\widehat{S}$, is consistent. ■

In this case, only type 1. restart clauses ($f(C) = \emptyset$).

Assume f is consistent wrt S, $|f(C)| \leq 1$, and there is exactly one restart clause.

$\Rightarrow$ SLWV-Resolution (Pereira et al., 1992)

Hyper Tableaux with Selection Function

Represent clauses by (non-Horn) rules:

$$\overbrace{L_1 \wedge \cdots \wedge L_m}^{\text{selected literals}} \rightarrow \overbrace{L_{m+1} \vee \cdots \vee L_n}^{\text{unselected literals}}$$

$\top \approx$ no selected literals, empty premiss

$\bot \approx$ only selected literals, empty conclusion

Example

f_N selects exactly the negative literals in a clause

$$S = \{\neg q \vee \neg s, \quad \neg r \vee s, \quad \overbrace{p \vee q \vee r}^{\text{Restart clause}}, \quad \neg p\}$$

$$\{q \wedge s \rightarrow \bot, \quad r \rightarrow s, \quad \overbrace{\top \rightarrow p \vee q \vee r}^{\text{Restart clause}}, \quad p \rightarrow \bot\}$$

Hyper Tableaux II

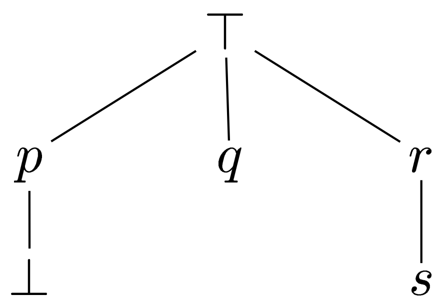
Which clauses C to use for extension of branch B?

- require $\mathbf{I}_B \not\models C$
- *all* selected literals in C weakly connected to B:
  $\overline{f(C)} \subseteq B$ ($f(C) = \emptyset$ admissible: restart clause)
- no restart clauses of type 2. (w/selected literals)

Example

Only unselected literals displayed. $\perp$ indicates closure.

$$\{q \wedge s \to \perp,\ r \to s,\ \overbrace{\top \to p \vee q \vee r}^{\text{Restart clause}},\ p \to \perp\}$$



Incomplete for general f (e.g. $f(C) = C$)
Complete, when f consistent.

Completeness of Hyper Tableaux

Theorem. For any unsatisfiable finite ground clause set S and consistent selection function f there is a hyper tableau proof with f for S. ■

Proof:

Assume to contrary there is a hyper tableau with f, open branch B, no more rule applicable.

S is finite $\leadsto$ B is finite.

$\widetilde{B}_f$ may be inconsistent, but:

B consistent; $\widetilde{B}_f - B$ consistent, because f consistent.

If $\widetilde{B}_f$ inconsistent there are literals
$L \in \{f(C) \mid C \in S, \mathbf{I}_B \not\models C\}$ s. th. $\overline{L}$ occurs in B.

B' obtained by removing all such literals from $\widetilde{B}_f$.

B' consistent — remains to show: $\mathbf{I}_{B'} \models S$.

Sufficient: $\mathbf{I}_{B'} \models C$ for C not used on B
(as $B' \supseteq B$, in particular $|f(C)| > 0$)

$\mathbf{I}_{B'} \not\models C$ would imply $\overline{f(C)} \subseteq B$
$\leadsto$ hyper extension step possible with C on B
↯ no more rule applicable on B ■

Positive Hyper Tableaux

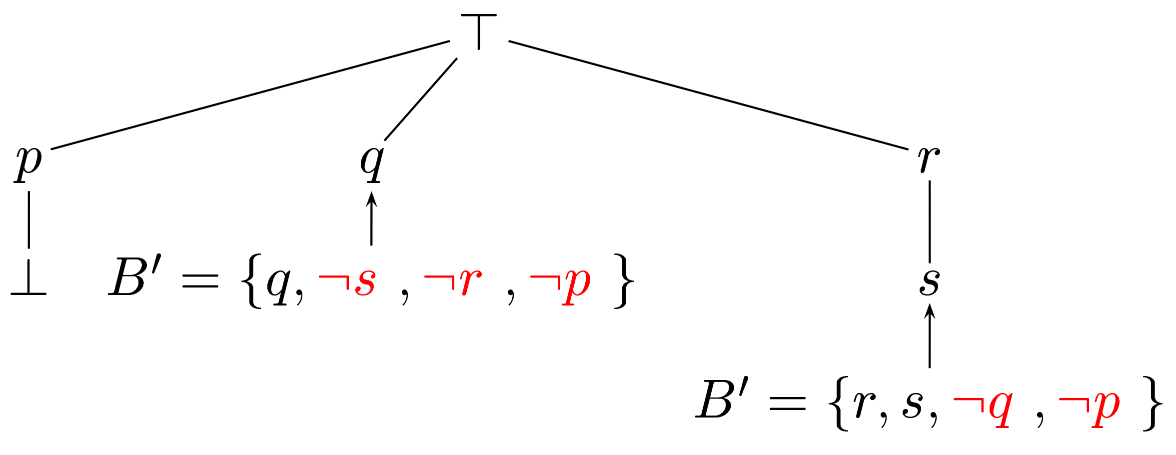
Positive hyper tableaux: fix f_N.

(e.g., Brown, 1978; Baumgartner et al., 1996f; Bry et al., 1996f)

Saturation simplifies to $B' = B \cup \{\neg p \mid p \in (\mathcal{A}_\Sigma - B)\}$

Example

$$\{q \wedge s \to \bot,\ \ r \to s,\ \ \top \to p \vee q \vee r,\ \ p \to \bot\}$$



Model Generation

(Manthey & Bry, 1988; Hasegawa et al., 1990f)

Definition.
A clause is **range-restricted** wrt a selection function f if $\mathsf{Var}(f(C)) \supseteq \mathsf{Var}(C - f(C))$. ∎

Given: clause set S range-restricted wrt f_N

$\mathcal{I}$ is set of Herbrand interpretations.

Interpretations (members of $\mathcal{I}$) called **model candidates**.

Init: Set $\mathcal{I}$ to $\{\emptyset\}$

Model Extension: $\mathbf{I}$ model candidate, $D \rightarrow E$ new instance of clause in S.

If there is a substitution σ such that $D\sigma \subseteq \mathbf{I}$ and $\mathbf{I} \not\models E\sigma$, then let $\mathcal{M}$

$$(\mathcal{M} - \{\mathbf{I}\}) \cup \bigcup_{L \in E\sigma} (\{L\} \cup \mathbf{I})$$

Model Rejection: $\mathbf{I}$ model candidate, $D \rightarrow \perp$ new instance of clause in S.

If there is a substitution σ such that $D\sigma \subseteq \mathbf{I}$, then delete $\mathbf{I}$ from $\mathcal{M}$.

Model Generation vs. Hyper Tableaux

Model generation $\approx$ positive hyper tableaux
for clauses range-restricted wrt f_N.

Model candidates $\approx$ open branches.

Model extension & model rejection $\approx$
special case of positive hyper tableau extension.

Tableau ground: MGU in extension step is matcher

All literals in tableau unselected, thus positive:
No reduction rule needed in positive hyper tableaux.

In general: no reduction, when no inconsistent unselected literals present.

Semantic Semantic Tableaux

Definition.
A selection function for S is **complete** if, for each ground atom p over the signature of S, it selects at least one literal q or $\neg q$ in S such that p is an instance of q. ■

Consistent & complete selection function $\approx$
total Herbrand interpretation

$\Rightarrow$ Semantic semantic tableaux

No reduction steps needed.

Application:

Constraint Model Generation (Shirai & Hasegawa, 1995)

$\approx$ hyper tableaux with selection function

$\Rightarrow$ CMG complete, if premisses consistent

Formal justification of CMG's completeness, e.g., when applied to QG problems.

Lifting Strategies

First-order clause tableaux

$\swarrow \qquad \searrow$

Destructive | Non-destructive

Destructive

- Incomplete search
  (Brown, 1978; Shults, 1997)
- Proof enumeration
  - Backtracking (Setheo, ProTein, $_3T^AP$)
  - Breadth-first search
- Strongly complete proof procedure

Non-destructive

- Fair rule application
  - ▶ Sentence/Smullyan tableaux
  - ▶ Tableaux with delayed closure
  - ▶ Hyper tableaux & range-restriction

Saturated Tableaux
(Bachmair & Ganzinger, 1999)

$$
\begin{array}{c}
\text{Resolution} \\
\big\downarrow \\
\text{Semantic Resolution} \\
\text{(Slagle, 1967)} \\
\big\downarrow \; [+\text{ splitting}] \\
\text{Semantic Semantic Tableaux}
\end{array}
$$

Making Clause Sets Consistent wrt any f

$$\cdots \overline{L'} \ \cdots \rightarrow \cdots \qquad\qquad \cdots \overline{L'} \ \cdots \rightarrow \cdots$$

$$\Rightarrow$$

$$L \wedge C \rightarrow D \qquad\qquad d(\vec{x}) \wedge C \rightarrow D \vee \overline{L}$$

L, $\overline{L'}$ unifiable

$\vec{x} = \mathsf{Var}(L) - \mathsf{Var}(C)$

d domain predicate of problem signature

- Inconsistency at L removed
- No new inconsistencies introduced
- Satisfiability & range-restrictedness preserved
- Problem size linear