# **KE/Tableaux**

The term *Tableaux* refers to a family of deduction methods for different logics. We start by introducing one of them:

"non-free-variable KE for classical FOL"

# What is it for?

Given:	set of premises $\Delta$ and conclusion $\varphi$ (all FOL sentences)		
Task:	prove $\Delta \models \varphi$		
How?	show $\Delta \cup \{\neg \varphi\}$ is not satisfiable (which is equivalent),		
	i.e. add the complement of the conclusion to the premises		
	and derive a contradiction ("refutation procedure")		

Ulle Endriss, King's College London

CS3AUR: Automated Reasoning 2002

KE/Tableaux

1

# **Constructing KE Proofs**

**Data structure.** A KE proof is represented as a *tableau*: a binary tree whose nodes are labelled with formulas.

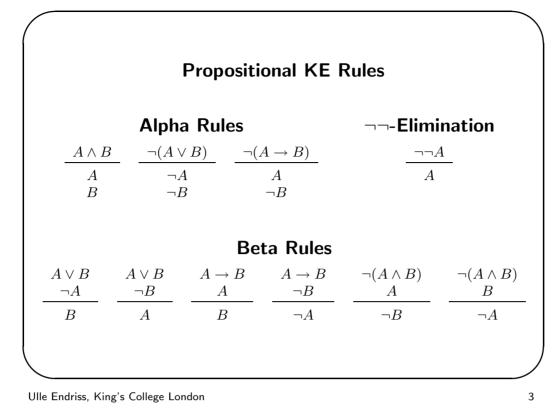
**Start.** We start by writing the premises and the negated conclusion into the root of an otherwise empty tableau.

**Expansion.** We apply *expansion rules* to the formulas on the tree; this results in new formulas being added and branches being split.

Closure. A branch that is obviously contradictory can be *closed*.

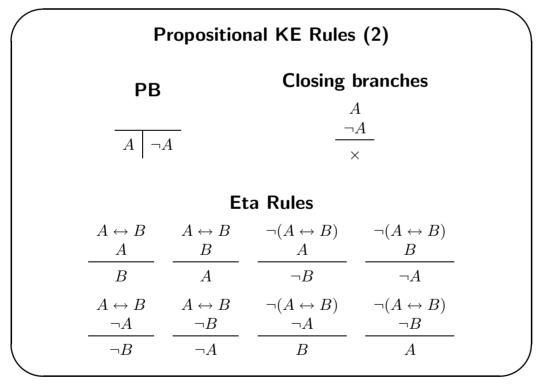
Success. A proof is *successful* iff we can close all branches.

Ulle Endriss, King's College London

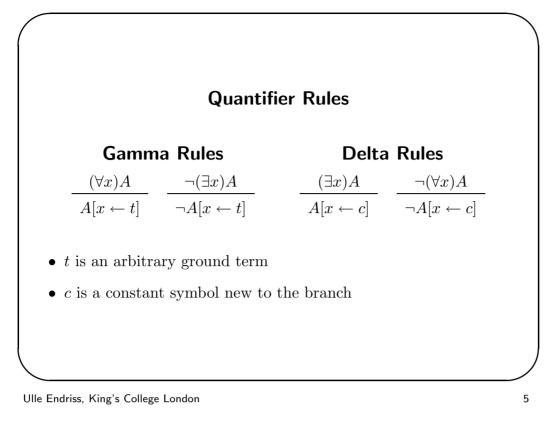


CS3AUR: Automated Reasoning 2002

KE/Tableaux

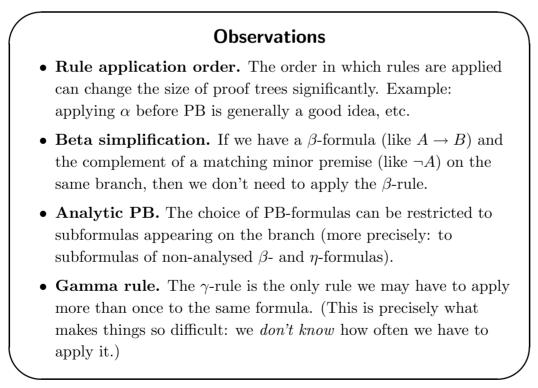


Ulle Endriss, King's College London



CS3AUR: Automated Reasoning 2002

KE/Tableaux



## Ulle Endriss, King's College London

# Can a proof ever fail?

**General answer:** No! The algorithm can only return "yes" or "don't know" (but not "no"). It is a so-called *semi-decision procedure* (there can be no decision procedure for FOL).

**Special cases.** In special cases, however, we may be able to "see" that a proof could never succeed (i.e. we can declare it a failure).

This is, for example, the case when we have enough information to construct a *countermodel* ...

Ulle Endriss, King's College London

CS3AUR: Automated Reasoning 2002

KE/Tableaux

7

# **Saturated Branches**

An open branch is called *saturated* iff every (complex) formulas has been analysed at least once and, additionally, every  $\gamma$ -formula has been instantiated with every term we can construct using the function symbols on the branch.

**Failing proofs.** A tableau with an open saturated branch can never be closed, i.e. we can stop an declare the proof a failure.

**The solution?** This only helps us in special cases though. (A single 1-ary function symbol together with a constant is already enough to construct infinitely many terms ...)

**Propositional logic.** In propositional logic (where we have no  $\gamma$ -formulas), after a limited number of steps, every branch will be either closed or saturated. This gives us a decision procedure.

Ulle Endriss, King's College London

# Countermodels

If a KE proof fails with a *saturated open branch*, you can use it to help you define a model  $\mathcal{M}$  for all the formulas on that branch:

- domain: set of all terms we can construct using the function symbols appearing on the branch (so-called *Herbrand universe*)
- terms are interpreted as themselves (*sic!*)
- interpretation of predicate symbols: see literals on branch

In particular,  $\mathcal{M}$  will be a model for the premises  $\Delta$  and the negated conclusion  $\neg \varphi$ , thereby constituting a *counterexample* for the attempted proof of  $\Delta \models \varphi$ .

**Careful:** There's a bug in WinKE: sometimes, what is presented as a countermodel is in fact only *part* of a countermodel (but it can always be extended to an actual model).

Ulle Endriss, King's College London

CS3AUR: Automated Reasoning 2002

KE/Tableaux

9

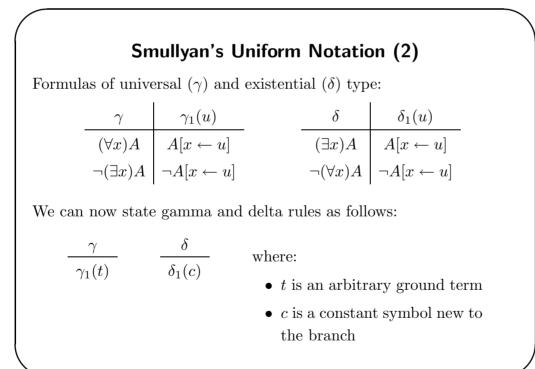
# **Soundness and Completeness** Again, let $\varphi$ be a sentence and let $\Delta$ be a set of sentences. We write $\Delta \vdash_{KE} \varphi$ to say that there exists a closed KE tableau for $\Delta \cup \{\neg \varphi\}$ . Before you can believe in KE you need to prove the following: **Theorem 1 (Soundness)** If $\Delta \vdash_{KE} \varphi$ then $\Delta \models \varphi$ . **Theorem 2 (Completeness)** If $\Delta \models \varphi$ then $\Delta \vdash_{KE} \varphi$ . **Important note:** The mere *existence* of a closed tableau does *not* entail that we have an effective method of finding it! Concretely: we don't know how often we need to apply the $\gamma$ -rule and what terms to use in the substitutions. **From now on,** to simplify things, we shall not consider $\rightarrow$ , $\leftrightarrow$ , $\top$ , and $\perp$ (which can be regarded as abbreviations).

Ulle Endriss, King's College London

Formul	as of conjund	ctive $(\alpha)$ and	disjunctive $(\beta)$ type:
1 orman	ů.		,
	$A \wedge B$	A B	$\begin{array}{ c c c c }\hline A \lor B & A & B \\\hline \end{array}$
	$\neg(A \lor B)$	$\neg A \neg B$	$\begin{array}{c cc} \beta & \beta_1 & \beta_2 \\ \hline A \lor B & A & B \\ \hline A \to B & \neg A & B \\ \neg (A \land B) & \neg A & \neg B \end{array}$
	$\neg (A \rightarrow B)$	$A \neg B$	$\neg (A \land B) \mid \neg A  \neg B$
		-	a rules as follows: where $A^{C} = \begin{cases} A' & \text{for } A = \neg A', \\ \neg A & \text{otherwise} \end{cases}$

CS3AUR: Automated Reasoning 2002

KE/Tableaux



Ulle Endriss, King's College London

# **Soundness Proof**

**Satisfiable branches.** We say that a *branch* is *satisfiable* iff the set of sentences on that branch is satisfiable.

**Proof sketch.** First prove the following lemma:

Lemma 1 (Satisfiable branches) If a non-branching KE rule is applied to a satisfiable branch, the result is another satisfiable branch. If PB is applied to a satisfiable branch, at least one of the resulting branches is also satisfiable.

Now we can prove soundness by contradiction: assume  $\Delta \vdash_{KE} \varphi$  but  $\Delta \not\models \varphi$  and try to derive a contradiction.

 $\Delta \not\models \varphi \implies \Delta \cup \{\neg\varphi\} \text{ satisfiable } \Rightarrow \text{ initial branch satisfiable} \\ \Rightarrow \text{ always at least one branch satisfiable (by above lemma)}$ 

This contradicts our assumption that at one point all branches will be closed  $(\Delta \vdash_{KE} \varphi)$ , because a closed branch is not satisfiable.

Ulle Endriss, King's College London

KE/Tableaux

13

# Hintikka's Lemma Definition 1 (Hintikka set) A set of sentences H is called a Hintikka set provided the following hold: (i) not both P ∈ H and ¬P ∈ H for propositional atoms P; (ii) if ¬¬φ ∈ H then φ ∈ H for all formulas φ; (iii) if α ∈ H then α<sub>1</sub> ∈ H and α<sub>2</sub> ∈ H for alpha formulas α; (iv) if β ∈ H then β<sub>1</sub> ∈ H or β<sub>2</sub> ∈ H for beta formulas β; (v) for all terms t constructible from function symbols in H (at least one constant symbol): if γ ∈ H then γ<sub>1</sub>(t) for gamma formulas γ; (vi) if δ ∈ H then δ<sub>1</sub>(t) ∈ H for some term t, for delta formulas δ.

Ulle Endriss, King's College London

# **Completeness Proof**

**Fairness.** We call a KE proof *fair* iff every (complex) formula gets *eventually* analysed on every branch and, additionally, every  $\gamma$ -formula gets *eventually* instantiated with every term constructible from the function symbols appearing on a branch.

**Proof sketch.** We will show the contrapositive: assume  $\Delta \not\models_{KE}$  and try to conclude  $\Delta \not\models \varphi$ .

If there is no KE proof for  $\Delta \cup \{\neg \varphi\}$  (assumption), then there can also be no *fair* KE proof. Show that a fairly constructed non-closable branch enumerates the elements of a Hintikka set H.

H is satisfiable (Hintikka's Lemma) and we have  $\Delta \subset H$  and  $\neg \varphi \in H$ .

So there is a model for  $\Delta \cup \{\neg\varphi\}$ , i.e. we get  $\Delta \not\models \varphi$ .

Ulle Endriss, King's College London

KE/Tableaux

15

# **Smullyan Tableaux**

The "standard" Tableaux rules (introduced by R. Smullyan in 1968) differ from the KE rules as follows:

- There is *no PB* rule.
- Beta rules are branching rules:

$$\begin{array}{c|c} A \lor B \\ \hline A & B \\ \hline \end{array} & \begin{array}{c|c} A \to B \\ \hline \neg A & B \\ \hline \hline \neg A & \neg B \\ \hline \end{array} & \begin{array}{c|c} \neg (A \land B) \\ \hline \neg A & \neg B \\ \hline \end{array} & \begin{array}{c|c} \text{in short:} & \begin{array}{c|c} \beta \\ \hline \beta_1 & \beta_2 \\ \hline \end{array} \end{array}$$

• Similarly for eta rules.

The rest is the same as with the KE system.

Ulle Endriss, King's College London