# Automated Theorem Proving

Georg Moser

Institute of Computer Science @ UIBK

Winter 2015

## Summary of Last Lecture

### Method of Davis and Putnam in Pseudo-Code

```
if $\mathcal{C}$ does not contain function symbols
then apply DPLL(a)-DPLL(c) on $\mathcal{C}'_0$
else {
  n := 0;
  contr := false;
  while (¬ contr) do {
    apply DPLL(a)-DPLL(c) on $\mathcal{C}'_n$;
    if the decision tree proves unsatisfiability,
    then contr := true
    else contr := false;
    n := n + 1;
  }}
```

### Definition

- individual constants
  $k_0, k_1, \dots, k_j, \dots$ denoted $c, d$, etc.
- function constants with $i$ arguments
  $f^i_0, f^i_1, \dots, f^i_j, \dots$ denoted $f, g, h$, etc.
- predicate constants with $i$ arguments
  $R^i_0, R^i_1, \dots, R^i_j, \dots$ denoted $P, Q, R$, etc.
- variables, collected in $\mathcal{V}$
  $x_0, x_1, \dots, x_j, \dots$ denoted $x$, $y$, $z$, etc.

### Definition

- propositional connectives $\neg$, $\vee$
- equality sign $=$

## Outline of the Lecture

### Early Approaches in Automated Reasoning

Herbrand's theorem for dummies, Gilmore's prover, method of Davis and Putnam

### Starting Points

resolution, tableau provers, Skolemisation, ordered resolution, redundancy and deletion

### Automated Reasoning with Equality

paramodulation, ordered completion and proof orders, superposition

### Applications of Automated Reasoning

Neuman-Stubblebinde Key Exchange Protocol, Robbins problem

## Resolution Calculus for First-Order Logic

restricted to atoms

### Definition

resolution

$$\frac{C \vee A \quad D \vee \neg B}{(C \vee D)\sigma}$$

factoring

$$\frac{C \vee A \vee B}{(C \vee A)\sigma}$$

$\sigma$ is a mgu of the atomic formulas $A$ and $B$

let $\mathcal{C}$ be a set of clauses; define resolution operator $\mathrm{Res}(\mathcal{C})$

- $\mathrm{Res}(\mathcal{C}) = \{D \mid D$ is resolvent or factor with premises in $\mathcal{C}\}$
- $\mathrm{Res}^0(\mathcal{C}) = \mathcal{C}$; $\mathrm{Res}^{n+1}(\mathcal{C}) = \mathrm{Res}^n(\mathcal{C}) \cup \mathrm{Res}(\mathrm{Res}^n(\mathcal{C}))$
- $\mathrm{Res}^*(\mathcal{C}) = \bigcup_{n \geqslant 0} \mathrm{Res}^n(\mathcal{C})$

### Example

$$\frac{\mathsf{P}(x) \vee \mathsf{Q}(\mathsf{f}(x, \mathsf{g}(y), x)) \quad \mathsf{R}(\mathsf{a}, \mathsf{b}) \vee \neg\mathsf{Q}(\mathsf{f}(z, \mathsf{g}(x'), \mathsf{h}(x')))}{\mathsf{P}(\mathsf{h}(x')) \vee \mathsf{R}(\mathsf{a}, \mathsf{b})} \; \{x \mapsto \mathsf{h}(x')\}$$

## Soundness and Completeness of Resolution

### Theorem

*resolution is sound: if $F$ a sentence and $\mathcal{C}$ its clause form such that $\square \in \mathrm{Res}^*(\mathcal{C})$, then $F$ is unsatisfiable*

### Proof.

- the theorem follows by case-distinction on the inferences
- for each inference one verifies that if the assumptions (as formulas) are modelled by an interpretation $\mathcal{M}$, then the consequence holds in $\mathcal{M}$ as well

■

### Theorem

*resolution is (refutationally) complete; if $F$ a sentence and $\mathcal{C}$ its clause form, then $\square \in \mathrm{Res}^*(\mathcal{C})$ if $F$ is unsatisfiable*

## Tableau Expansion Rules

### Definition (uniform notation)

| conjunctive | | | disjunctive | | |
|---|---|---|---|---|---|
| $\alpha$ | $\alpha_1$ | $\alpha_2$ | $\beta$ | $\beta_1$ | $\beta_2$ |
| $A \wedge B$ | $A$ | $B$ | $\neg(A \wedge B)$ | $\neg A$ | $\neg B$ |
| $\neg(A \vee B)$ | $\neg A$ | $\neg B$ | $A \vee B$ | $A$ | $B$ |
| $\neg(A \rightarrow B)$ | $A$ | $\neg B$ | $A \rightarrow B$ | $\neg A$ | $B$ |

### Definition (tableau expansion rules)

$$\frac{\neg\neg A}{A} \qquad \frac{\alpha}{\begin{array}{c}\alpha_1 \\ \alpha_2\end{array}} \qquad \frac{\beta}{\beta_1 \mid \beta_2}$$

## Reminder: Propositional Semantic Tableaux

Computational Logic: week 3

### Definition

let $\{A_1, \ldots, A_n\}$ be propositional formulas

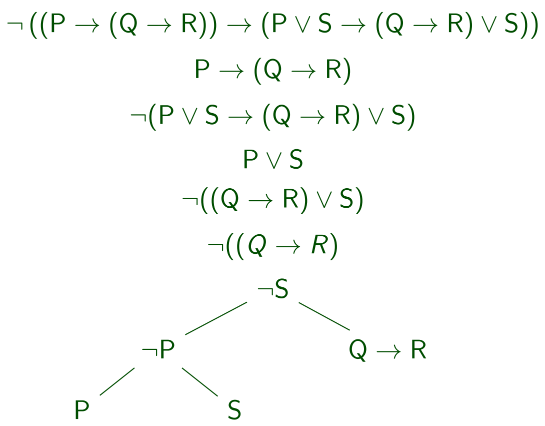
- the following tree $T$ is a tableau for $\{A_1, \ldots, A_n\}$:

$$\begin{array}{c} A_1 \\ A_2 \\ \vdots \\ A_n \end{array}$$

- suppose $T$ is a tableau for $\{A_1, \ldots, A_n\}$ and $T^*$ is obtained by applying a tableau expansion rule to $T$, then $T^*$ is a tableau for $\{A_1, \ldots, A_n\}$

### Example

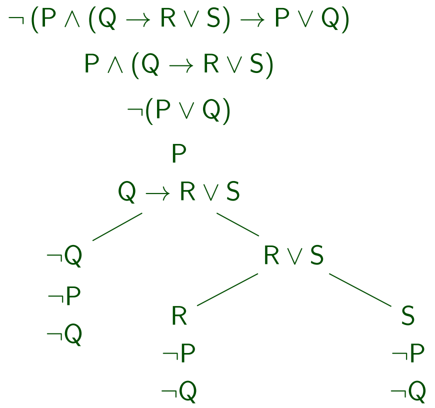
consider the tableau proof of $(P \to (Q \to R)) \to (P \lor S \to (Q \to R) \lor S)$


$$\neg\,((P \to (Q \to R)) \to (P \lor S \to (Q \to R) \lor S))$$
$$P \to (Q \to R)$$
$$\neg(P \lor S \to (Q \to R) \lor S)$$
$$P \lor S$$
$$\neg((Q \to R) \lor S)$$
$$\neg((Q \to R)$$
$$\neg S$$

## Heuristics Matters

### Example

consider $P \land (Q \to R \lor S) \to P \lor Q$ and the following tableau proof


$$\neg\,(P \land (Q \to R \lor S) \to P \lor Q)$$
$$P \land (Q \to R \lor S)$$
$$\neg(P \lor Q)$$
$$P$$
$$Q \to R \lor S$$

### Example (cont'd)

now consider the following tableau proof

$$\neg\,((P \land (Q \to R \lor S)) \to P \lor Q)$$
$$P \land (Q \to R \lor S)$$
$$\neg(P \lor Q)$$
$$P$$
$$Q \to R \lor S$$
$$\neg P$$
$$\neg Q$$

## Soundness and Completeness

### Definitions

- a branch is closed if the formulas $F$ and $\neg F$ occur on it
- if $F$ is atomic, then the branch is said to be atomically closed
- a tableau is closed if every branch is closed
- a tableau proof of $F$ is a closed tableau for $\neg F$
- in a strict tableau no formula is expanded twice on the same branch

### Theorem

*the tableau procedure is sound and complete:*

$$F \text{ is a tautology} \iff F \text{ has a tableau proof}$$

### Proof.

use next two lemmas; alternative proof of completeness: propositional model existence lemma ■

## Strong Propositional Completeness

### Lemma

*any application of a tableau expansion rule to a satisfiable tableau yields another satisfiable tableau*

### Lemma

*suppose $F$ is a valid; a strict tableau construction for $\neg F$ that is continued as long as possible must terminate in an atomically closed tableau*

### Proof.

see Computational Logic, this week ■

## Implementation of Semantic Tableaux

### Naive Approach

```
tableau_prover(X) :-
        expand([[neg X]],Y),
        closed(Y).
```

### Slightly More Efficient

```
tableau_prover2(X) :-
        expand([[neg X]],Y),
        !,
        closed(Y).
```

### A Bit More Efficient

```
tableau_prover3(X) :-
        expand_and_close([[neg X]]).
```

## First-Order Semantic Tableaux

### Definition (uniform notation)

| universal | | existential | |
|---|---|---|---|
| $\gamma$ | $\gamma(t)$ | $\delta$ | $\delta(t)$ |
| $\forall x A(x)$ | $A(t)$ | $\exists x A(x)$ | $A(t)$ |
| $\neg \exists x A(x)$ | $\neg A(t)$ | $\neg \forall x A(x)$ | $\neg A(t)$ |

### Definition (tableau expansion rules)

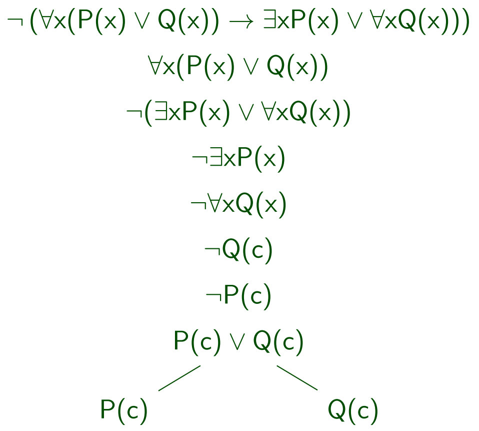
$$\frac{\gamma}{\gamma(t)} \quad t \text{ term in } \mathcal{L}^+ \qquad \frac{\delta}{\delta(k)} \quad k \text{ fresh constant in } \mathcal{L}^+$$

1. $\mathcal{L}^+$ denotes extension of base language $\mathcal{L}$
2. new individual constants are introduced in $\delta$ rules
3. fresh means new to the branch of the tableau

### Example

consider $\forall x(P(x) \lor Q(x)) \to \exists x P(x) \lor \forall x Q(x)$

we give a tableau proof



# Soundness and Completeness of Tableau

## Definitions

- a tableau proof of a sentence $F$ is a closed tableau for $\neg F$
- a tableau branch is satisfiable if the set $\mathcal{G}$ of sentences on it is satisfiable, i.e., there exists a model of $\mathcal{G}$; a tableau is satisfiable if some branch is satisfiable

## Theorem

*if sentence $F$ has a tableau proof, then $F$ is valid*

## Proof.

if any tableau expansion rule is applied to a satisfiable tableau, the result is satisfiable ■

## Theorem

*if a sentence $F$ is valid, then $F$ has a tableau proof*