Mathematical Logic

Tableaux Reasoning for Propositional Logic

Chiara Ghidini

FBK-IRST, Trento, Italy

Outline of this lecture

- An introduction to Automated Reasoning with Analytic Tableaux;
- Today we will be looking into tableau methods for classical propositional logic (we will discuss first-order tableaux later).
- Analytic Tableaux are a a family of mechanical proof methods, developed for a variety of different logics. Tableaux are nice, because they are both easy to grasp for *humans* and easy to implement on *machines*.

Tableaux

- Early work by Beth and Hintikka (around 1955). Later refined and popularised by Raymond Smullyan:
  - R.M. Smullyan. First-order Logic. Springer-Verlag, 1968.
- Modern expositions include:
  - M. Fitting. First-order Logic and Automated Theorem Proving. 2nd edition. Springer-Verlag, 1996.
  - M. DAgostino, D. Gabbay, R. Hähnle, and J. Posegga (eds.). Handbook of Tableau Methods. Kluwer, 1999.
  - R. Hähnle. Tableaux and Related Methods. In: A. Robinson and A. Voronkov (eds.), Handbook of Automated Reasoning, Elsevier Science and MIT Press, 2001.
  - Proceedings of the yearly Tableaux conference: `http://i12www.ira.uka.de/TABLEAUX/`

How does it work?

The tableau method is a method for proving, in a mechanical manner, that a given set of formulas is not satisfiable. In particular, this allows us to perform automated *deduction*:

Given : set of premises Γ and conclusion ϕ

Task : prove $\Gamma \models \phi$

How does it work?

The tableau method is a method for proving, in a mechanical manner, that a given set of formulas is not satisfiable. In particular, this allows us to perform automated *deduction*:

Given :	set of premises Γ and conclusion ϕ
Task :	prove $\Gamma \models \phi$
How?	show $\Gamma \cup \neg\phi$ is not satisfiable (which is equivalent),
	i.e. add the complement of the conclusion to the premises
	and derive a contradiction (refutation procedure)

Reduce Logical Consequence to (un)Satisfiability

Theorem

$\Gamma \models \phi$ *if and only if* $\Gamma \cup \{\neg\phi\}$ *is unsatisfiable*

Proof.

$\Rightarrow$ Suppose that $\Gamma \models \phi$, this means that every interpretation $\mathcal{I}$ that satisfies Γ, it does satisfy ϕ, and therefore $\mathcal{I} \not\models \neg\phi$. This implies that there is no interpretations that satisfies together Γ and $\neg\phi$.

$\Leftarrow$ Suppose that $\mathcal{I} \models \Gamma$, let us prove that $\mathcal{I} \models \phi$, Since $\Gamma \cup \{\neg\phi\}$ is not satisfiable, then $\mathcal{I} \not\models \neg\phi$ and therefore $\mathcal{I} \models \phi$.

$\square$

Reduce Logical Consequence to (un)Satisfiability

Theorem

$\Gamma \models \phi$ *if and only if* $\Gamma \cup \{\neg\phi\}$ *is unsatisfiable*

Proof.

$\Rightarrow$ Suppose that $\Gamma \models \phi$, this means that every interpretation $\mathcal{I}$ that satisfies Γ, it does satisfy ϕ, and therefore $\mathcal{I} \not\models \neg\phi$. This implies that there is no interpretations that satisfies together Γ and $\neg\phi$.

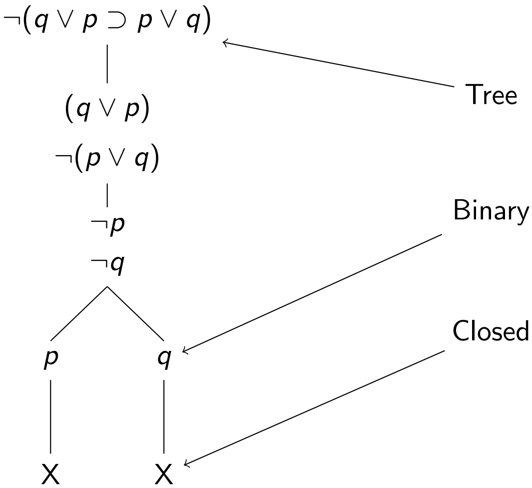
$\Leftarrow$ Suppose that $\mathcal{I} \models \Gamma$, let us prove that $\mathcal{I} \models \phi$, Since $\Gamma \cup \{\neg\phi\}$ is not satisfiable, then $\mathcal{I} \not\models \neg\phi$ and therefore $\mathcal{I} \models \phi$.

$\square$

Constructing Tableau Proofs

- **Data structure**: a proof is represented as a tableau - i.e., a binary tree - the nodes of which are labelled with formulas.
- **Start**: we start by putting 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, thereby adding new formulas and splitting branches.
- **Closure**: we close branches that are obviously contradictory.
- **Success**: a proof is successful iff we can close all branches.

An example


$\neg(q \vee p \supset p \vee q)$
$(q \vee p)$
$\neg(p \vee q)$
$\neg p$
$\neg q$
p
q
X
X
Tree
Binary
Closed

Expansion Rules of Propositional Tableau

α rules

$$\frac{\phi \wedge \psi}{\begin{array}{c}\phi \\ \psi\end{array}} \qquad \frac{\neg(\phi \vee \psi)}{\begin{array}{c}\neg\phi \\ \neg\psi\end{array}} \qquad \frac{\neg(\phi \supset \psi)}{\begin{array}{c}\phi \\ \neg\psi\end{array}}$$

$\neg\neg$-Elimination

$$\frac{\neg\neg\phi}{\phi}$$

β rules

$$\frac{\phi \vee \psi}{\phi \mid \psi} \qquad \frac{\neg(\phi \wedge \psi)}{\neg\phi \mid \neg\psi} \qquad \frac{\phi \supset \psi}{\neg\phi \mid \psi}$$

Branch Closure

$$\frac{\begin{array}{c}\phi \\ \neg\phi\end{array}}{X}$$

Note: These are the standard ("Smullyan-style") tableau rules.

We omit the rules for $\equiv$. We rewrite $\phi \equiv \psi$ as $(\phi \supset \psi) \wedge (\psi \supset \phi)$

Smullyans Uniform Notation

Two types of formulas: conjunctive (α) and disjunctive (β):

α	α_1	α_2
$\phi \wedge \psi$	ϕ	ψ
$\neg(\phi \vee \psi)$	$\neg\phi$	$\neg\psi$
$\neg(\phi \supset \psi)$	ϕ	$\neg\psi$

β	β_1	β_2
$\phi \vee \psi$	ϕ	ψ
$\neg(\phi \wedge \psi)$	$\neg\phi$	$\neg\psi$
$\phi \supset \psi$	$\neg\phi$	ψ

We can now state α and β rules as follows:

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

Note: α rules are also called deterministic rules. β rules are also called splitting rules.

Smullyans Uniform Notation

Two types of formulas: conjunctive (α) and disjunctive (β):

α	α_1	α_2
$\phi \wedge \psi$	ϕ	ψ
$\neg(\phi \vee \psi)$	$\neg\phi$	$\neg\psi$
$\neg(\phi \supset \psi)$	ϕ	$\neg\psi$

β	β_1	β_2
$\phi \vee \psi$	ϕ	ψ
$\neg(\phi \wedge \psi)$	$\neg\phi$	$\neg\psi$
$\phi \supset \psi$	$\neg\phi$	ψ

We can now state α and β rules as follows:

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

Note: α rules are also called deterministic rules. β rules are also called splitting rules.

An example

$$\neg(q \lor p \supset p \lor q)$$

An example

$$\neg(q \vee p \supset p \vee q)$$

$$(q \vee p)$$

$$\neg(p \vee q)$$

An example

$$\neg(q \vee p \supset p \vee q)$$

$$(q \vee p)$$

$$\neg(p \vee q)$$

$$\neg p$$

$$\neg q$$

An example

$$\neg(q \vee p \supset p \vee q)$$

$$(q \vee p)$$

$$\neg(p \vee q)$$

$$\neg p$$

$$\neg q$$

$$\begin{array}{cc} p & \qquad q \end{array}$$

An example

$$\neg(q \vee p \supset p \vee q)$$

$$(q \vee p)$$

$$\neg(p \vee q)$$

$$\neg p$$

$$\neg q$$

p	q
X	X

Some definition for tableaux

Definition (type-*alpha* and type-β formulae)

- Formulae of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, and $\neg(\phi \supset \psi)$ are called type-α formulae.
- Formulae of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, and $\phi \supset \psi$ are called type-β formulae

Note: type-*alpha* formulae are the ones where we use α rules. type-β formulae are the ones where we use β rules.

Definition (Closed branch)

A closed branch is a branch which contains a formula and its negation.

Definition (Open branch)

An open branch is a branch which is not closed

Definition (Closed tableaux)

A tableaux is closed if all its branches are closed.

Definition (Derivation $\Gamma \vdash \phi$)

Let ϕ and Γ be a propositional formula and a finite set of propositional formulae, respectively. We write $\Gamma \vdash \phi$ to say that there exists a closed tableau for $\Gamma \cup \{\neg\phi\}$

Some definition for tableaux

Definition (type-*alpha* and type-β formulae)

- Formulae of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, and $\neg(\phi \supset \psi)$ are called type-α formulae.
- Formulae of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, and $\phi \supset \psi$ are called type-β formulae

Note: type-*alpha* formulae are the ones where we use α rules. type-β formulae are the ones where we use β rules.

Definition (Closed branch)

A closed branch is a branch which contains a formula and its negation.

Definition (Open branch)

An open branch is a branch which is not closed

Definition (Closed tableaux)

A tableaux is closed if all its branches are closed.

Definition (Derivation $\Gamma \vdash \phi$)

Let ϕ and Γ be a propositional formula and a finite set of propositional formulae, respectively. We write $\Gamma \vdash \phi$ to say that there exists a closed tableau for $\Gamma \cup \{\neg\phi\}$

Some definition for tableaux

Definition (type-*alpha* and type-β formulae)

- Formulae of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, and $\neg(\phi \supset \psi)$ are called type-α formulae.
- Formulae of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, and $\phi \supset \psi$ are called type-β formulae

Note: type-*alpha* formulae are the ones where we use α rules. type-β formulae are the ones where we use β rules.

Definition (Closed branch)

A closed branch is a branch which contains a formula and its negation.

Definition (Open branch)

An open branch is a branch which is not closed

Definition (Closed tableaux)

A tableaux is closed if all its branches are closed.

Definition (Derivation $\Gamma \vdash \phi$)

Let ϕ and Γ be a propositional formula and a finite set of propositional formulae, respectively. We write $\Gamma \vdash \phi$ to say that there exists a closed tableau for $\Gamma \cup \{\neg\phi\}$

Some definition for tableaux

Definition (type-*alpha* and type-β formulae)

- Formulae of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, and $\neg(\phi \supset \psi)$ are called type-α formulae.
- Formulae of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, and $\phi \supset \psi$ are called type-β formulae

Note: type-*alpha* formulae are the ones where we use α rules. type-β formulae are the ones where we use β rules.

Definition (Closed branch)

A closed branch is a branch which contains a formula and its negation.

Definition (Open branch)

An open branch is a branch which is not closed

Definition (Closed tableaux)

A tableaux is closed if all its branches are closed.

Definition (Derivation $\Gamma \vdash \phi$)

Let ϕ and Γ be a propositional formula and a finite set of propositional formulae, respectively. We write $\Gamma \vdash \phi$ to say that there exists a closed tableau for $\Gamma \cup \{\neg\phi\}$

Exercises

Exercise

Show that the following are valid arguments:

- $\models ((P \supset Q) \supset P) \supset P$
- $P \supset (Q \wedge R), \neg Q \vee \neg R \models \neg P$

Solutions

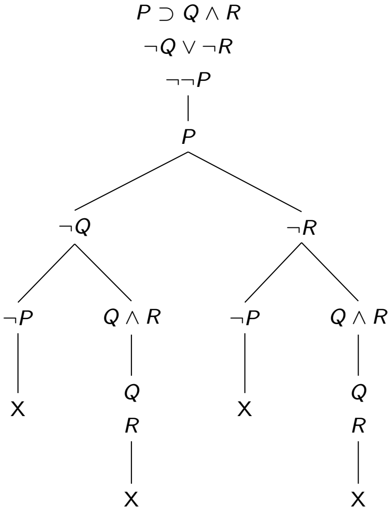
$$\neg(((P \supset Q) \supset P) \supset P)$$

$$(P \supset Q) \supset P$$

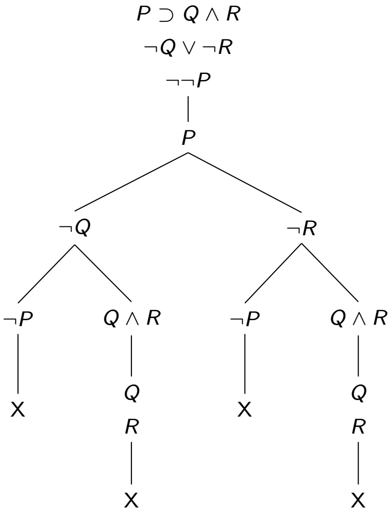
$$\neg P$$

$$\begin{array}{ccc} \neg(P \supset Q) & \qquad & P \\ P & & \times \\ \neg Q & & \\ \times & & \end{array}$$

Solutions


$P \supset Q \wedge R$
$\neg Q \vee \neg R$
$\neg\neg P$
P
$\neg Q$
$\neg R$
$\neg P$
$Q \wedge R$
$\neg P$
$Q \wedge R$
X
Q
R
X
X
Q
R
X

Solutions



Note: different orderings of expansion rules are possible! But all lead to unsatisfiability.

Tableaux and satisfiability

- A tableau for Γ attempts to build a propositional interpretation for Γ. If the tableaux is closed, it means that no model exist.
- We can use tableaux to check if a formula is satisfiable.

Exercise

Check whether the formula $\neg((P \supset Q) \wedge (P \wedge Q \supset R) \supset (P \supset R))$ is satisfiable

Tableaux and satisfiability

- A tableau for Γ attempts to build a propositional interpretation for Γ. If the tableaux is closed, it means that no model exist.
- We can use tableaux to check if a formula is satisfiable.

Exercise

Check whether the formula $\neg((P \supset Q) \wedge (P \wedge Q \supset R) \supset (P \supset R))$ is satisfiable

Tableaux and satisfiability

- A tableau for Γ attempts to build a propositional interpretation for Γ. If the tableaux is closed, it means that no model exist.
- We can use tableaux to check if a formula is satisfiable.

Exercise

Check whether the formula $\neg((P \supset Q) \wedge (P \wedge Q \supset R) \supset (P \supset R))$ is satisfiable

Solution

$$\neg((P \supset Q) \wedge (P \wedge Q \supset R) \supset (P \supset R))$$

$$(P \supset Q) \wedge (P \wedge Q \supset R)$$

$$\neg(P \supset R)$$

$$P \supset Q$$

$$P \wedge Q \supset R$$

$$P$$

$$\neg R$$

$$\begin{array}{ccccc} \neg P & & & Q & \\ \mid & & & & \\ \text{X} & & \neg(P \wedge Q) & & R \\ & & & & \mid \\ & \neg P & & \neg Q & \text{X} \\ & \mid & & \mid & \\ & \text{X} & & \text{X} & \end{array}$$

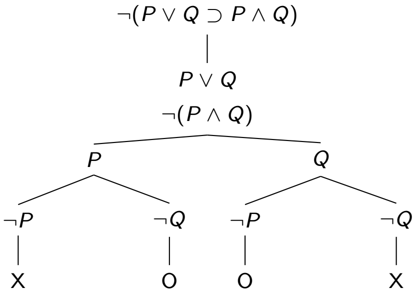
The tableau is closed and the formula is not satisfiable.

Satisfiability: An example

Exercise

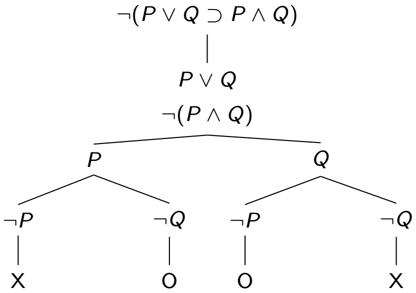
Check whether the formula $\neg(P \vee Q \supset P \wedge Q)$ is satisfiable

Solution



Two open branches. The formula is satisfiable.
The tableau shows us all the possible interpretations ($\{P\}, \{Q\}$) that satisfy the formula.

Solution



Two open branches. The formula is satisfiable.
The tableau shows us all the possible interpretations $(\{P\}, \{Q\})$ that satisfy the formula.

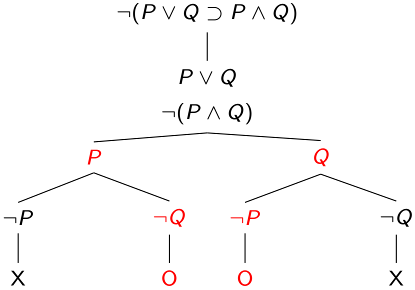
Using the tableau to build interpretations.

For each open branch in the tableau, and for each propositional atom p in the formula we define

$$\mathcal{I}(p) = \begin{cases} \text{True} & \text{if } p \text{ belongs to the branch,} \\ \text{False} & \text{if } \neg p \text{ belongs to the branch.} \end{cases}$$

If neither p nor $\neg p$ belong to the branch we can define $\mathcal{I}(p)$ in an arbitrary way.

Models for $\neg(P \vee Q \supset P \wedge Q)$



Two models:

- $\mathcal{I}(P) = \text{True}, \mathcal{I}(Q) = \text{False}$
- $\mathcal{I}(P) = \text{False}, \mathcal{I}(Q) = \text{True}$

Double-check with the truth tables!

P	Q	$P \vee Q$	$P \wedge Q$	$P \vee Q \supset P \wedge Q$	$\neg(P \vee Q \supset P \wedge Q)$
T	T	T	T	T	F
F	F	F	F	T	F
T	F	T	F	F	T
F	T	T	F	F	T

Double-check with the truth tables!

P	Q	$P \vee Q$	$P \wedge Q$	$P \vee Q \supset P \wedge Q$	$\neg(P \vee Q \supset P \wedge Q)$
T	T	T	T	T	F
F	F	F	F	T	F
T	F	T	F	F	T
F	T	T	F	F	T

Homeworks!

Exercise

Show *unsatisfiability* of each of the following formulae using tableaux:

- $(p \equiv q) \equiv (\neg q \equiv p)$;
- $\neg((\neg q \supset \neg p) \supset ((\neg q \supset p) \supset q))$.

Show *satisfiability* of each of the following formulae using tableaux:

- $(p \equiv q) \supset (\neg q \equiv p)$;
- $\neg(p \vee q \supset ((\neg p \wedge q) \vee p \vee \neg q))$.

Show *validity* of each of the following formulae using tableaux:

- $(p \supset q) \supset ((p \supset \neg q) \supset \neg p)$;
- $(p \supset r) \supset (p \vee q \supset r \vee q)$.

For each of the following formulae, *describe all models* of this formula using tableaux:

- $(q \supset (p \wedge r)) \wedge \neg(p \vee r \supset q)$;
- $\neg((p \supset q) \wedge (p \wedge q \supset r) \supset (\neg p \supset r))$.

Establish the *equivalences* between the following pairs of formulae using tableaux:

- $(p \supset \neg p), \neg p$;
- $(p \supset q), (\neg q \supset \neg p)$;
- $(p \vee q) \wedge (p \vee \neg q), p$.

Termination

Assuming we analyse each formula at most once, we have:

Theorem (Termination)

For any propositional tableau, after a finite number of steps no more expansion rules will be applicable.

Hint for proof: This must be so, because each rule results in ever shorter formulas.

Note: Importantly, termination will *not* hold in the first-order case.

Preliminary definition

Definition (Literal)

A literal is an atomic formula p or the negation $\neg p$ of an atomic formula.

Termination

Hint of proof:

Base case Assume that we have a literal formula. Then it is a propositional variable or a negation of a propositional variable and no expansion rules are applicable.

Inductive step Assume that the theorem holds for any formula with at most n connectives and prove it with a formula θ with $n+1$ connectives.
Three cases:

- θ is a type-α formula (of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, or $\neg(\phi \supset \psi)$)

  We have to apply an α-rule

  $$\begin{array}{c} \theta \\ | \\ \alpha_1 \\ \alpha_2 \end{array}$$

  and we mark the formula θ as analysed once.

  Since α_1 and α_2 contain less connectives than θ we can apply the inductive hypothesis and say that we can build a propositional tableau such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

  α_1, α_2

  We concatenate the two trees and the proof is done.

Hint of proof:

Base case Assume that we have a literal formula. Then it is a propositional variable or a negation of a propositional variable and no expansion rules are applicable.

Inductive step Assume that the theorem holds for any formula with at most n connectives and prove it with a formula θ with $n+1$ connectives.
Three cases:

- θ is a type-α formula (of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, or $\neg(\phi \supset \psi)$)

  We have to apply an α-rule

  $$\begin{array}{c} \theta \\ | \\ \alpha_1 \\ \alpha_2 \end{array}$$

  and we mark the formula θ as analysed once.

  Since α_1 and α_2 contain less connectives than θ we can apply the inductive hypothesis and say that we can build a propositional tableau such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

  α_1, α_2

  We concatenate the two trees and the proof is done.

Termination

Hint of proof:

Base case Assume that we have a literal formula. Then it is a propositional variable or a negation of a propositional variable and no expansion rules are applicable.

Inductive step Assume that the theorem holds for any formula with at most n connectives and prove it with a formula θ with $n+1$ connectives.
Three cases:

- θ is a type-α formula (of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, or $\neg(\phi \supset \psi)$)

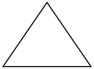
  We have to apply an α-rule

$$\begin{array}{c} \theta \\ | \\ \alpha_1 \\ \alpha_2 \end{array}$$

  and we mark the formula θ as analysed once.

  Since α_1 and α_2 contain less connectives than θ we can apply the inductive hypothesis and say that we can build a propositional tableau such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

α_1, α_2



  We concatenate the two trees and the proof is done.

Termination

Hint of proof:

Base case Assume that we have a literal formula. Then it is a propositional variable or a negation of a propositional variable and no expansion rules are applicable.

Inductive step Assume that the theorem holds for any formula with at most n connectives and prove it with a formula θ with $n+1$ connectives.
Three cases:

- θ is a type-α formula (of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, or $\neg(\phi \supset \psi)$)

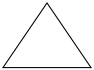
  We have to apply an α-rule

  $$\begin{array}{c} \theta \\ | \\ \alpha_1 \\ \alpha_2 \end{array}$$

  and we mark the formula θ as analysed once.

  Since α_1 and α_2 contain less connectives than θ we can apply the inductive hypothesis and say that we can build a propositional tableau such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

  α_1, α_2

  We concatenate the two trees and the proof is done.

Termination

Three cases:

- θ is a type-β formula (of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, or $\phi \supset \psi$)

  We have to apply a β-rule

  and we mark the formula θ as analysed once.

  Since β_1 and β_2 contain less connectives than θ we can apply the inductive hypothesis and say that we can build two propositional tableaux, one for β_1 and one for β_2 such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

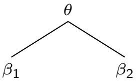
  We concatenate the 3 trees and the proof is done.

Termination

Three cases:

- θ is a type-β formula (of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, or $\phi \supset \psi$)

  We have to apply a β-rule



  and we mark the formula θ as analysed once.

  Since β_1 and β_2 contain less connectives than θ we can apply the inductive hypothesis and say that we can build two propositional tableaux, one for β_1 and one for β_2 such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

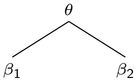
  We concatenate the 3 trees and the proof is done.

Termination

Three cases:

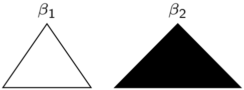
- θ is a type-β formula (of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, or $\phi \supset \psi$)

  We have to apply a β-rule

  and we mark the formula θ as analysed once.

  Since β_1 and β_2 contain less connectives than θ we can apply the inductive hypothesis and say that we can build two propositional tableaux, one for β_1 and one for β_2 such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

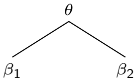
  We concatenate the 3 trees and the proof is done.

Termination

Three cases:

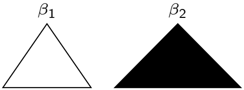
- θ is a type-β formula (of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, or $\phi \supset \psi$)

  We have to apply a β-rule

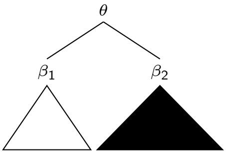


  and we mark the formula θ as analysed once.

  Since β_1 and β_2 contain less connectives than θ we can apply the inductive hypothesis and say that we can build two propositional tableaux, one for β_1 and one for β_2 such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.



  We concatenate the 3 trees and the proof is done.



Termination

- θ is of the form $\neg\neg\phi$.

  We have to apply the $\neg\neg$-Elimination rule

$$\begin{array}{c} \neg\neg\phi \\ | \\ \phi \end{array}$$

  and we mark the formula $\neg\neg\phi$ as analysed once.

  Since ϕ contains less connectives than $\neg\neg\phi$ we can apply the inductive hypothesis and say that we can build a propositional tableaux for it such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

ϕ

  We concatenate the 2 trees and the proof is done.

Termination

- θ is of the form $\neg\neg\phi$.

  We have to apply the $\neg\neg$-Elimination rule

  $$\begin{array}{c} \neg\neg\phi \\ | \\ \phi \end{array}$$

  and we mark the formula $\neg\neg\phi$ as analysed once.

  Since ϕ contains less connectives than $\neg\neg\phi$ we can apply the inductive hypothesis and say that we can build a propositional tableaux for it such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

  We concatenate the 2 trees and the proof is done.

Termination

- θ is of the form $\neg\neg\phi$.

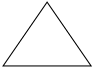
  We have to apply the $\neg\neg$-Elimination rule

$$\begin{array}{c} \neg\neg\phi \\ | \\ \phi \end{array}$$

  and we mark the formula $\neg\neg\phi$ as analysed once.

  Since ϕ contains less connectives than $\neg\neg\phi$ we can apply the inductive hypothesis and say that we can build a propositional tableaux for it such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

ϕ



  We concatenate the 2 trees and the proof is done.

Termination

- θ is of the form $\neg\neg\phi$.

  We have to apply the $\neg\neg$-Elimination rule

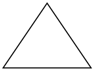
  $$\neg\neg\phi$$

  $$\phi$$

  and we mark the formula $\neg\neg\phi$ as analysed once.

  Since ϕ contains less connectives than $\neg\neg\phi$ we can apply the inductive hypothesis and say that we can build a propositional tableaux for it such that each formula is analysed at most once and after a finite number of steps no more expansion rules will be applicable.

  ϕ



  We concatenate the 2 trees and the proof is done.

To actually believe that the tableau method is a valid decision procedure we have to prove:

Theorem (Soundness)

If $\Gamma \vdash \phi$ *then* $\Gamma \models \phi$

Theorem (Completeness)

If $\Gamma \models \phi$ *then* $\Gamma \vdash \phi$

Remember: We write $\Gamma \vdash \phi$ to say that there exists a closed tableau for $\Gamma \cup \{\neg\phi\}$.

Proof of Soundness - preliminary definitions

Definition (Saturated propositional tableau)

A branch of a propositional tableau is saturated if all the (non-literal) formulae occurring in the branch have been analysed. A tableau is saturated if all its branches are saturated.

Definition (Satisfiable branch)

A branch β of a tableaux τ is satisfiable if the set of formulas that occurs in β is satisfiable. I.e., if there is an interpretation $\mathcal{I}$, such that $\mathcal{I} \models \phi$ for all $\phi \in \beta$.

Proof of Soundness - preliminary definitions

Definition (Saturated propositional tableau)

A branch of a propositional tableau is saturated if all the (non-literal) formulae occurring in the branch have been analysed. A tableau is saturated if all its branches are saturated.

Definition (Satisfiable branch)

A branch β of a tableaux τ is satisfiable if the set of formulas that occurs in β is satisfiable. I.e., if there is an interpretation $\mathcal{I}$, such that $\mathcal{I} \models \phi$ for all $\phi \in \beta$.

Proof of Soundness - preliminary lemma

First prove the following lemma:

Lemma (Satisfiable Branches)

- *If a non-branching rule is applied to a satisfiable branch, the result is another satisfiable branch.*
- *If a branching rule is applied to a satisfiable branch, at least one of the resulting branches is also satisfiable.*

Hint for proof: prove for all the expansion rules that they extend a satisfiable branch *sb* to (at least) a branch sb' which is satisfiable.

Proof of Soundness - proof of preliminary lemma

Propositional α-rules: the example of $\wedge$

$$\frac{\phi \wedge \psi}{\begin{array}{c}\phi \\ \psi\end{array}}$$

- let $\mathcal{I}$ be such that $\mathcal{I} \models sb$
- since $\phi \wedge \psi \in sb$ then $\mathcal{I} \models \phi \wedge \psi$
- which implies that $\mathcal{I} \models \phi$ and $\mathcal{I} \models \psi$
- which implies that $\mathcal{I} \models sb'$ with $sb' = sb \cup \{\phi, \psi\}$.

Propositional β-rules: the example of $\vee$

$$\frac{\phi \vee \psi}{\phi \mid \psi}$$

- let $\mathcal{I}$ be such that $\mathcal{I} \models sb$
- since $\phi \vee \psi \in sb$ then $\mathcal{I} \models \phi \vee \psi$
- which implies that $\mathcal{I} \models \phi$ or $\mathcal{I} \models \psi$
- which implies that $\mathcal{I} \models sb'$ with $sb' = sb \cup \{\phi\}$ or $\mathcal{I} \models sb''$ with $sb'' = sb \cup \{\psi\}$.

Proof of Soundness (II)

We have to show that $\Gamma \vdash \phi$ implies $\Gamma \models \phi$. We prove it by contradiction, that is, assume $\Gamma \vdash \phi$ but $\Gamma \not\models \phi$ and try to derive a contradiction.

- If $\Gamma \not\models \phi$ then $\Gamma \cup \{\neg\phi\}$ is satisfiable (see theorem on relation between logical consequence and (un) satisfiability)
- therefore the initial branch of the tableau (the root $\Gamma \cup \{\neg\phi\}$) is satisfiable
- therefore the tableau for this formula will always have a satisfiable branch (see previouls Lemma on satisfiable branches)
- This contradicts our assumption that at one point all branches will be closed ($\Gamma \vdash \phi$), because a closed branch clearly is not satisfiable.
- Therefore we can conclude that $\Gamma \not\models \phi$ cannot be and therefore that $\Gamma \models \phi$ holds.

Proof of Completeness - the Hintikkas Lemma

Definition (Hintikka set)

A set of propositional formulas Γ is called a Hintikka set provided the following hold:

1. not both $p \in H$ and $\neg p \in H$ for all propositional atoms p;
2. if $\neg\neg\phi \in H$ then $\phi \in H$ for all formulas ϕ ;
3. if $\phi \in H$ and ϕ is a type-α formula then $\alpha_1 \in H$ and $\alpha_2 \in H$;
4. if $\phi \in H$ and ϕ is a type-β formula then either $\beta_1 \in H$ or $\beta_2 \in H$.

Remember:

- type-α formulae are of the form $\phi \wedge \psi$, $\neg(\phi \vee \psi)$, or $\neg(\phi \supset \psi)$
- type-β formulae are of the form $\phi \vee \psi$, $\neg(\phi \wedge \psi)$, or $\phi \supset \psi$

Lemma (Hintikka Lemma)

Every Hintikka set is satisfiable

Proof:

- We construct a model $\mathcal{I} : \mathcal{P} \rightarrow \{\text{True}, \text{False}\}$ from a given Hintikka set H as follows:
  Let $\mathcal{P}$ be the set of propositional variables occurring in literals of H,

$$\mathcal{I}(p) = \begin{cases} \text{True} & \text{if } p \in H, \\ \text{False} & \text{if } p \notin H. \end{cases}$$

- We now prove that $\mathcal{I}$ is a propositional model that satisfies all the formulae in H. That is, if $\phi \in H$ then $\mathcal{I} \models \phi$.

  Base case We investigate literal formulae.
  Let p be an atomic formula in H. Then $\mathcal{I}(p) = \mathit{True}$ by definition of $\mathcal{I}$. Thus, $\mathcal{I} \models p$
  Let $\neg p$ be a negation of an atomic formula in H. From the property (1) of Hintikka set, the fact that $\neg p$ belongs to H implies that $p \notin H$. Therefore from the definition of $\mathcal{I}$ we have that $\mathcal{I}(p) = \mathit{False}$, and therefore $\mathcal{I} \models \neg p$

Proof of Completeness - Hintikkas Lemma (c'nd)

Inductive step We prove the theorem for all non-literal formulae.

- Let θ be of the form $\neg\neg\phi$.
  Then because of the property (2) of Hintikka sets $\phi \in H$.
  Therefore $\mathcal{I} \models \phi$ because of the inductive hypothesis.
  Then $\mathcal{I} \not\models \neg\phi$ and $\mathcal{I} \models \neg\neg\phi$ because of the definition of propositonal satisfiability of $\neg$.
- Let θ be a type-α formula. Then, its components α_1 and α_2 belong to H because of property (3) of the Hintikka set.
  We can apply the inductive hypothesis to α_1 and α_2 and derive that $\mathcal{I} \models \alpha_1$ and $\mathcal{I} \models \alpha_2$
  It is now easy to prove that $\mathcal{I} \models \theta$
- Let θ be a type-β formula. Then, at least one of its components β_1 or β_2 belong to H because of property (4) of the Hintikka set.
  We can apply the inductive hypothesis to β_1 or β_2 and derive that $\mathcal{I} \models \beta_1$ or $\mathcal{I} \models \beta_2$
  It is now easy to prove that $\mathcal{I} \models \theta$

Proof of Completeness - Hintikkas Lemma (c'nd)

Inductive step We prove the theorem for all non-literal formulae.

- Let θ be of the form $\neg\neg\phi$.
  Then because of the property (2) of Hintikka sets $\phi \in H$.
  Therefore $\mathcal{I} \models \phi$ because of the inductive hypothesis.
  Then $\mathcal{I} \not\models \neg\phi$ and $\mathcal{I} \models \neg\neg\phi$ because of the definition of propositonal satisfiability of $\neg$.
- Let θ be a type-α formula. Then, its components α_1 and α_2 belong to H because of property (3) of the Hintikka set.
  We can apply the inductive hypothesis to α_1 and α_2 and derive that $\mathcal{I} \models \alpha_1$ and $\mathcal{I} \models \alpha_2$
  It is now easy to prove that $\mathcal{I} \models \theta$
- Let θ be a type-β formula. Then, at least one of its components β_1 or β_2 belong to H because of property (4) of the Hintikka set.
  We can apply the inductive hypothesis to β_1 or β_2 and derive that $\mathcal{I} \models \beta_1$ or $\mathcal{I} \models \beta_2$
  It is now easy to prove that $\mathcal{I} \models \theta$

Proof of Completeness - Hintikkas Lemma (c'nd)

Inductive step We prove the theorem for all non-literal formulae.

- Let θ be of the form $\neg\neg\phi$.
  Then because of the property (2) of Hintikka sets $\phi \in H$.
  Therefore $\mathcal{I} \models \phi$ because of the inductive hypothesis.
  Then $\mathcal{I} \not\models \neg\phi$ and $\mathcal{I} \models \neg\neg\phi$ because of the definition of propositonal satisfiability of $\neg$.
- Let θ be a type-α formula. Then, its components α_1 and α_2 belong to H because of property (3) of the Hintikka set.
  We can apply the inductive hypothesis to α_1 and α_2 and derive that $\mathcal{I} \models \alpha_1$ and $\mathcal{I} \models \alpha_2$
  It is now easy to prove that $\mathcal{I} \models \theta$
- Let θ be a type-β formula. Then, at least one of its components β_1 or β_2 belong to H because of property (4) of the Hintikka set.
  We can apply the inductive hypothesis to β_1 or β_2 and derive that $\mathcal{I} \models \beta_1$ or $\mathcal{I} \models \beta_2$
  It is now easy to prove that $\mathcal{I} \models \theta$

A last definition - Fairness

Definition (Fairness)

We call a propositional tableau fair if every non-literal of a branch gets eventually analysed on this branch.

Proof of Completeness

Completeness proof (sketch).

- We show that $\Gamma \nvdash \phi$ implies $\Gamma \not\models \phi$.
- Suppose that there is no proof for $\Gamma \cup \{\neg\phi\}$
- Let τ a fair tableaux that start with $\Gamma \cup \{\neg\phi\}$,
- The fact that $\Gamma \nvdash \phi$ implies that there is at least an open branch *ob*.
- fairness condition implies that the set of formulas in *ob* constitute an Hintikka set H_{ob}
- From Hintikka lemma we have that there is an interpretation $\mathcal{I}_{ob}$ that satisfies *ob*.
- since every branch of τ contains its root we have that $\Gamma \cup \{\neg\phi\} \subseteq ob$ and therefore $\mathcal{I}_{ob} \models \Gamma \cup \{\neg\phi\}$.
- which implies that $\Gamma \not\models \phi$.

$\square$

Decidability

The proof of Soundness and Completeness confirms the decidability of propositional logic:

Theorem (Decidability)

The tableau method is a decision procedure for classical propositional logic.

Proof. To check validity of ϕ, develop a tableau for $\neg\phi$. Because of termination, we will eventually get a tableau that is either (1) closed or (2) that has a branch that cannot be closed.

- In case (1), the formula ϕ must be valid (soundness).
- In case (2), the branch that cannot be closed shows that $\neg\phi$ is satisfiable (see completeness proof), i.e. ϕ cannot be valid.

This terminates the proof.

Exercise

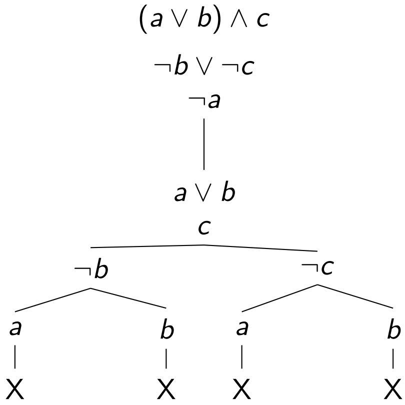
Exercise

Build a tableau for $\{(a \vee b) \wedge c, \neg b \vee \neg c, \neg a\}$

Exercise

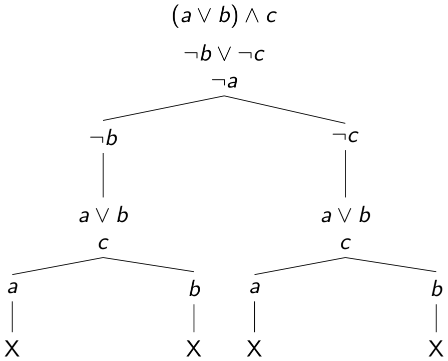
Exercise

Build a tableau for $\{(a \vee b) \wedge c, \neg b \vee \neg c, \neg a\}$



Another solution

What happens if we first expand the disjunction and then the conjunction?



Expanding β rules creates new branches. Then α rules may need to be expanded in all of them.

Strategies of expansion

- Using the “wrong” policy (e.g., expanding disjunctions first) leads to an increase of *size* of the tableau, which leads to an increase of *time*;
- yet, unsatisfiability is still proved if set is unsatisfiable;
- this is not the case for other logics, where applying the wrong policy may inhibit proving unsatisfiability of some unsatisfiable sets.

Finding Short Proofs

- It is an open problem to find an efficient algorithm to decide in all cases which rule to use next in order to derive the shortest possible proof.
- However, as a rough guideline always apply any applicable *non-branching rules* first. In some cases, these may turn out to be redundant, but they will never cause an exponential blow-up of the proof.

Efficiency

- Are analytic tableaus an efficient method of checking whether a formula is a tautology?
- Remember: using the truth-tables to check a formula involving n propositional atoms requires filling in 2^n rows (exponential = very bad).
- Are tableaux any better?
- In the worst case no, but if we are lucky we may skip some of the 2^n rows !!!

Efficiency

- Are analytic tableaus an efficient method of checking whether a formula is a tautology?
- Remember: using the truth-tables to check a formula involving n propositional atoms requires filling in 2^n rows (exponential = very bad).
- Are tableaux any better?
- In the worst case no, but if we are lucky we may skip some of the 2^n rows !!!

Efficiency

- Are analytic tableaus an efficient method of checking whether a formula is a tautology?
- Remember: using the truth-tables to check a formula involving n propositional atoms requires filling in 2^n rows (exponential $=$ very bad).
- Are tableaux any better?
- In the worst case no, but if we are lucky we may skip some of the 2^n rows !!!

Efficiency

- Are analytic tableaus an efficient method of checking whether a formula is a tautology?
- Remember: using the truth-tables to check a formula involving n propositional atoms requires filling in 2^n rows (exponential = very bad).
- Are tableaux any better?
- In the worst case no, but if we are lucky we may skip some of the 2^n rows !!!

Exercise

Exercise

Give proofs for the unsatisfiability of the following formula using (1) truth-tables, and (2) Smullyan-style tableaux.

$$(P \vee Q) \wedge (P \vee \neg Q) \wedge (\neg P \vee Q) \wedge (\neg P \vee \neg Q)$$