Integrating Computation in Logic: Deduction Modulo

Olivier Hermant

28 September 2007
Deduction and Computation

- Computation is at the root of mathematics.
Deduction and Computation

- Computation is at the root of mathematics.
- It has been forgotten by the formalization of the mathematics.
Deduction and Computation

- Computation is at the root of mathematics.
- It has been forgotten by the formalization of the mathematics.
- Reborn with informatics: rewriting rules.
- We need a balance between deduction steps and computation steps.
Deduction systems: the logical framework

- first-order logic: function and predicate symbols, logical connectors: $\land$, $\lor$, $\Rightarrow$, $\neg$, and quantifiers $\forall$, $\exists$.

$\forall n (\text{Even}(n) \Rightarrow \text{Odd}(n + 1))$
$\forall n (\text{Odd}(n) \Rightarrow \text{Even}(n + 1))$

$\text{Even}(0)$
Deduction systems: the logical framework

- first-order logic: function and predicate symbols, logical connectors: $\land, \lor, \Rightarrow, \neg$, and quantifiers $\forall, \exists$.

\[
\begin{align*}
\text{Even}(0) \\
\forall n (\text{Even}(n) \Rightarrow \text{Odd}(n+1)) \\
\forall n (\text{Odd}(n) \Rightarrow \text{Even}(n+1))
\end{align*}
\]

- a sequent:

\[
\begin{array}{c}
\text{hyp.} \\
\Gamma \vdash A
\end{array}
\]

- rules to form them: sequent calculus (or natural deduction)
- framework: intuitionistic logic (classical, linear, higher-order, constraints ...)

Deduction System : sequents calculus (LJ)

- A deduction rule:

\[
\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \land B}
\]

- right and left rules

\[
\begin{align*}
\Gamma, A \vdash A & \quad \text{axiom} \\
\Gamma, A \vdash A & \quad \text{axiom} \\
\Gamma \vdash A \quad \Gamma \vdash B & \quad \Lambda \land -r \\
\Gamma \vdash A \land B & \\
\Gamma, A, B \vdash C & \quad \text{cut} \\
\Gamma \vdash B & \quad \text{cut} \\
\Gamma, A, B \vdash C & \quad \Lambda \land -l \\
\Gamma, A \land B \vdash C & \\
\Gamma \vdash A[t] & \quad \text{\texttt{V-g}, any } t \\
\Gamma, \forall x A[x], A[t] \vdash B & \\
\Gamma, \forall x A[x] \vdash B & \\
\Gamma \vdash A[x] & \quad \text{\texttt{V-r}, x free} \\
\Gamma \vdash \forall x A[x] & \\
\end{align*}
\]
Example: 1

\[ \forall x P(x) \vdash P(0) \land P(1) \]
Example: 1

\[ \forall x P(x) \vdash P(0) \quad \forall x P(x) \vdash P(1) \]

\[ \frac{\forall x P(x) \vdash P(0) \land P(1)}{\forall x P(x) \vdash P(0) \land P(1)} \quad \land \text{-}r \]
Example: 1

\[
\forall x \mathcal{P}(x), P(0) \vdash P(0) \\
\forall x \mathcal{P}(x) \vdash P(0)
\]

\[
\forall x \mathcal{P}(x), P(1) \vdash P(1) \\
\forall x \mathcal{P}(x) \vdash P(1)
\]

\[
\forall x \mathcal{P}(x) \vdash P(0) \land P(1)
\]

\[
\forall -I \quad \forall -I \quad \land -r
\]
Example: 1

\[ \forall x P(x), P(0) \vdash P(0) \quad \text{axiom} \]
\[ \forall x P(x) \vdash P(0) \]
\[ \forall x P(x) \vdash P(0) \land P(1) \quad \wedge\text{-r} \]

\[ \forall x P(x), P(1) \vdash P(0) \quad \text{axiom} \]
\[ \forall x P(x) \vdash P(1) \]
\[ \forall x P(x) \vdash P(0) \land P(1) \quad \wedge\text{-r} \]
Example: 2

\[ \forall x P(x) \vdash P(0) \land P(1) \]
Example: 2

\[\forall x P(x), P(1), P(0) \vdash P(0) \land P(1)\]

\[\forall x P(x), P(0) \vdash P(0) \land P(1)\]

\[\forall x P(x) \vdash P(0) \land P(1)\]
Example: 2

\[ \forall x P(x), P(1), P(0) \vdash P(0) \quad \forall x P(x), P(1), P(0) \vdash P(1) \]

\[ \forall x P(x), P(1), P(0) \vdash P(0) \land P(1) \]

\[ \forall x P(x), P(0) \vdash P(0) \land P(1) \]

\[ \forall x P(x) \vdash P(0) \land P(1) \]

Axiom, \land -r

Axiom

\land -l

\forall -l

\forall -l
Example: 2

\[
\begin{align*}
\text{axiom} & \quad \forall x P(x), P(1), P(0) \vdash P(0) \quad \forall x P(x), P(1), P(0) \vdash P(1) \\
\forall x P(x), P(1), P(0) \vdash P(0) \land P(1) & \quad \forall x P(x), P(1), P(0) \vdash P(0) \land P(1) \\
\forall x P(x), P(0) \vdash P(0) \land P(1) & \quad \forall x P(x) \vdash P(0) \land P(1)
\end{align*}
\]

- the first rule is not always “don’t care”: free variable condition.
### Axioms vs. rewriting

<table>
<thead>
<tr>
<th>Axioms</th>
<th>Rewriting</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x + S(y) = S(x + y)$</td>
<td>$x + S(y) \rightarrow S(x + y)$</td>
</tr>
<tr>
<td>$x + 0 = x$</td>
<td>$x + 0 \rightarrow x$</td>
</tr>
<tr>
<td>$x \times 0 = 0$</td>
<td>$x \times 0 \rightarrow 0$</td>
</tr>
<tr>
<td>$x \times S(y) = x + x \times y$</td>
<td>$x \times S(y) \rightarrow x + x \times y$</td>
</tr>
<tr>
<td>$(x \times y = 0) \Leftrightarrow (x = 0 \lor y = 0)$</td>
<td>$(x \times y = 0) \rightarrow (x = 0 \lor y = 0)$</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\vdash 2 \times 2 = 4$</td>
<td>$\vdash 4 = 4$</td>
</tr>
<tr>
<td>$\vdash \exists x(2 \times x = 4)$</td>
<td>$\vdash \exists x(2 \times x = 4)$</td>
</tr>
</tbody>
</table>
Deduction modulo: allowed rewriting

- General form (free variables are possible):
  \[ l \rightarrow r \]

  - use: We replace \( t = \sigma l \) by \( \sigma r \) (unification). Rewriting could be deep in the term.
  - rewriting on terms:
    \[ x + S(y) \rightarrow S(x + y) \]

  - and on propositions (predicate symbols):
    \[ x \ast y = 0 \rightarrow x = 0 \lor y = 0 \]

  - advantage: expressiveness

  - we obtain a congruence modulo \( R \) (chosen set of rules):
    \[ \equiv \]

  - deduction rules transform as such:
    \[ \text{axiom } \Gamma, A \vdash A \text{ becomes } \text{axiom, } A \equiv B \Gamma, A \vdash B \]
Deduction modulo: allowed rewriting

- General form (free variables are possible):
  \[ l \rightarrow r \]
- use: We replace \( t = \sigma l \) by \( \sigma r \) (unification). Rewriting could be deep in the term.
Deduction modulo: allowed rewriting

- General form (free variables are possible):
  \[ l \rightarrow r \]

- use: We replace \( t = \sigma l \) by \( \sigma r \) (unification). Rewriting could be deep in the term.

- rewriting on terms:
  \[ x + S(y) \rightarrow S(x + y) \]
Deduction modulo: allowed rewriting

- General form (free variables are possible):
  \[ l \rightarrow r \]

- Use: We replace \( t = \sigma l \) by \( \sigma r \) (unification). Rewriting could be deep in the term.

- Rewriting on terms:
  \[ x + S(y) \rightarrow S(x + y) \]

- And on **propositions** (predicate symbols):
  \[ x \ast y = 0 \rightarrow x = 0 \lor y = 0 \]

- Advantage: expressiveness
Deduction modulo: allowed rewriting

- General form (free variables are possible):
  \[ l \rightarrow r \]

- use: We replace \( t = \sigma l \) by \( \sigma r \) (unification). Rewriting could be deep in the term.

- rewriting on terms:
  \[ x + S(y) \rightarrow S(x + y) \]

- and on **propositions** (predicate symbols):
  \[ x \ast y = 0 \rightarrow x = 0 \lor y = 0 \]

- advantage: expressiveness

- we obtain a congruence modulo \( R \) (chosen set of rules): \( \equiv \)
Deduction modulo: allowed rewriting

- General form (free variables are possible):
  \[ l \rightarrow r \]
- use: We replace \( t = \sigma l \) by \( \sigma r \) (unification). Rewriting could be deep in the term.
- rewriting on terms:
  \[ x + S(y) \rightarrow S(x + y) \]
- and on propositions (predicate symbols):
  \[ x \ast y = 0 \rightarrow x = 0 \lor y = 0 \]
- advantage: expressiveness
- we obtain a congruence modulo \( \mathcal{R} \) (chosen set of rules): \( \equiv \)
- deduction rules transform as such:
  \[
  \frac{\text{axiom}}{\Gamma, A \vdash A}
  \] becomes
  \[
  \frac{\text{axiom, } A \equiv B}{\Gamma, A \vdash B}
  \]
Deduction modulo : sequent calculus modulo

\[ \Gamma, A \vdash B \]\n\[ \text{axiom } A \equiv B \]

\[ \Gamma \vdash A \quad \Gamma \vdash B \]
\[ \frac{\Gamma \vdash C}{\Gamma \vdash A \land B \equiv C} \land \text{-r} \]

\[ \Gamma, B, A[t] \vdash C \]
\[ \frac{\Gamma, B \vdash C}{\Gamma \vdash \forall x A[x] \equiv B} \forall \text{-l} \]

\[ \Gamma, A \vdash C \quad \Gamma \vdash B \]
\[ \frac{\Gamma \vdash C}{\Gamma \vdash C \land -r \ A \land B \equiv D} \land \text{-l} \]

\[ \Gamma, A, B \vdash C \]
\[ \frac{\Gamma, D \vdash C}{\Gamma \vdash A \land B \equiv D} \land \text{-r} \]

\[ \Gamma \vdash A[x] \]
\[ \frac{\Gamma \vdash B}{\Gamma \vdash \forall x A[x] \equiv B} \forall \text{-r} \]

\[ \Gamma, B \vdash C \]
\[ \frac{\Gamma \vdash C}{\Gamma \vdash B} \forall \text{-r} \]
Example: 3

consider the rewriting system $\mathcal{R}$:

$P(0) \rightarrow A$

$P(1) \rightarrow B$

$\forall x P(x) \vdash A \land B$
Example: 3

consider the rewriting system $R$:

$$P(0) \rightarrow A$$
$$P(1) \rightarrow B$$

$$\forall x P(x) \vdash A \quad \forall x P(x) \vdash B$$

$$\frac{\forall x P(x) \vdash A \quad \forall x P(x) \vdash B}{\forall x P(x) \vdash A \land B} \land\text{-}r$$
Example: 3

- consider the rewriting system $\mathcal{R}$:

\[
P(0) \rightarrow A
\]

\[
P(1) \rightarrow B
\]

\[
\forall x P(x) \vdash A \quad \forall x P(x), P(0) \vdash A
\]

\[
\forall x P(x) \vdash A \quad \forall x P(x), P(1) \vdash B
\]

\[
\forall x P(x) \vdash B \quad \forall x P(x) \vdash A \land B
\]

\[
\forall x P(x) \vdash A \land B
\]
Example: 3

▶ consider the rewriting system \( R \):

\[
P(0) \rightarrow A \\
P(1) \rightarrow B
\]

\[
\text{axiom} \quad \forall x P(x), P(0) \vdash B \\
\quad \forall x P(x) \vdash A
\]

\[
\forall x P(x), P(1) \vdash B \\
\quad \forall x P(x) \vdash B
\]

\[
\text{axiom} \quad \forall x P(x) \vdash A \land B
\]
Cut rule: a detour

\[
\frac{\Gamma, A \vdash B \quad \Gamma \vdash C}{\Gamma \vdash B} \text{ cut, } A \equiv C
\]

- show $\Gamma \vdash A$
- show $\Gamma, A \vdash B$
- then, you have showed $\Gamma \vdash B$
- it is the application of a lemma.
Example: 4

Consider the rewriting system $\mathcal{R}$:

- $P(0) \rightarrow A$
- $P(1) \rightarrow B$

$\forall x P(x) \vdash A \land B$
Example: 4

- consider the rewriting system $\mathcal{R}$:

  \[
  \begin{align*}
  P(0) & \rightarrow A \\
  P(1) & \rightarrow B
  \end{align*}
  \]

  $\forall x P(x), A \vdash A \land B$ \quad $\forall x P(x) \vdash A$

  \[
  \begin{array}{c}
  \forall x P(x) \vdash A \\
  \vdash A \land B
  \end{array}
  \]

  
  \[
  \text{cut}
  \]
Example: 4

Consider the rewriting system $\mathcal{R}$:

$$
\begin{align*}
P(0) & \to A \\
P(1) & \to B
\end{align*}
$$

$$
\begin{align*}
\forall x P(x), A & \vdash A \land B \\
\forall x P(x), P(0) & \vdash A \\
\forall x P(x) & \vdash A \\
\forall x P(x), A & \vdash A \land B
\end{align*}
$$

\[\forall x P(x) \vdash A \land B \]
Example: 4

- consider the rewriting system $\mathcal{R}$:

\[
P(0) \rightarrow A
\]
\[
P(1) \rightarrow B
\]

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B \\
\forall x P(x), A \vdash A \land B \\
\forall x P(x) \vdash A \land B
\end{array}
\]

- cut

\[
\begin{array}{c}
\forall x P(x), P(1), A \vdash B \\
\forall x P(x), P(0) \vdash A \\
\forall x P(x) \vdash A \land B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), P(1), A \vdash B \\
\forall x P(x) \vdash A \land B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B \\
\forall x P(x), A \vdash A \land B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x) \vdash A \land B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x) \vdash A \land B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B
\end{array}
\]

- $\forall$-r

\[
\begin{array}{c}
\forall x P(x), A \vdash A \\
\forall x P(x), A \vdash B
\end{array}
\]

- $\forall$-r
Example: 4

- consider the rewriting system $\mathcal{R}$:

$$\begin{align*}
P(0) & \rightarrow A \\
P(1) & \rightarrow B
\end{align*}$$

$$\begin{align*}
\forall x P(x), A & \vdash A \\
\forall x P(x), A & \vdash B \\
\forall x P(x), A & \vdash A \land B
\end{align*}$$

$$\begin{align*}
\forall x P(x), P(0) & \vdash A \\
\forall x P(x) & \vdash A \land B \\
\forall x P(x) & \vdash A \land B
\end{align*}$$

- an unnecessary detour
- we could have cutted on any formula!
The cut rule: a detour

\[
\frac{\Gamma, A \vdash B \quad \Gamma \vdash C}{\Gamma \vdash B} \quad \text{cut} \quad A \equiv C
\]

- we show $\Gamma, A \vdash B$ and $\Gamma \vdash A$
- then we have showed $\Gamma \vdash B$.
- lemma: the good way for a human being.
- in practice: not adapted for automatic demonstration.
  Nb: resolution method do not proceed by cuts!
The cut rule: a detour

\[ \frac{\Gamma, A \vdash B \quad \Gamma \vdash C}{\Gamma \vdash B} \quad \text{cut} \quad A \equiv C \]

- we show \( \Gamma, A \vdash B \) and \( \Gamma \vdash A \)
- then we have showed \( \Gamma \vdash B \).
- lemma: the good way for a human being.
- in practice: not adapted for automatic demonstration.
  Nb: resolution method *do not* proceed by cuts!
- in theory: consistence, proof normalization (Curry-Howard) depend of its elimination.
The cut rule: a detour

\[
\frac{\Gamma, A \vdash B \quad \Gamma \vdash C}{\Gamma \vdash B} \text{ cut } A \equiv C
\]

- we show \(\Gamma, A \vdash B\) and \(\Gamma \vdash A\)
- then we have showed \(\Gamma \vdash B\).
- lemma: the good way for a human being.
- in practice: not adapted for automatic demonstration.
  Nb: resolution method *do not* proceed by cuts!
- in theory: consistence, proof normalization (Curry-Howard) depend of its elimination.
- eliminating cuts: a key result.

\[
\Gamma \vdash A \quad \Gamma \vdash_{cf} A
\]

- two main paths towards:
  - proof normalization (syntactic).
  - semantical methods.
The cut rule: a detour

\[
\frac{\Gamma, A \vdash B \quad \Gamma \vdash C}{\Gamma \vdash B} \text{ cut } A \equiv C
\]

- we show \( \Gamma, A \vdash B \) and \( \Gamma \vdash A \)
- then we have showed \( \Gamma \vdash B \).
- lemma: the good way for a human being.
- in practice: not adapted for automatic demonstration.
  Nb: resolution method \textit{do not} proceed by cuts!
- in theory: consistence, proof normalization (Curry-Howard) depend of its elimination.
- eliminating cuts: a key result.

\[
\Gamma \vdash A \triangleright \Gamma \vdash_{cf} A
\]

- two main paths towards:
  - proof normalization (syntactic).
  - semantical methods.
- in deduction modulo: indecidable, need for general criterions on \( R \)
The normalization method(s)

- Curry-Howard: proofs = programs
- formulas = types
- proof tree = typing tree
- at the heart of proof assistants (PVS, Coq, Isabelle, ...)
- when a program calculates, it performs a cut elimination procedure.
The normalization method(s)

- Curry-Howard: proofs = programs
- formulas = types
- proof tree = typing tree
- at the heart of proof assistants (PVS, Coq, Isabelle, ...)
- when a program calculates, it performs a cut elimination procedure.
- show that all typables function terminates.
The semantical method(s)

- define a semantical space (truth value). Ex: Boolean algebras.
- we must have soundness/completeness wrt the semantic.
The semantical method

\[ \Gamma \vdash A \quad \text{soundness} \quad \Gamma \models A \]

Gentzen
Tait-Girard
Dowek-Werner
...

\[ \Gamma \vdash_{cf} A \quad \text{completeness} \]
The semantical method

\[ \Gamma \vdash A \quad \text{soundness} \quad \Gamma \models A \]

Gentzen
Tait-Girard
Dowek-Werner

\[ \Gamma \vdash_{cf} A \quad \text{completeness} \quad \Gamma \models A \]

\[ \Gamma \vdash A \quad \text{strong completeness} \]

...
The semantical method

\[ \Gamma \vdash A \quad \text{soundness} \quad \Gamma \models A \]

\[ \Gamma \vdash_{cf} A \quad \text{strong completeness} \]

Gentzen
Tait-Girard
Dowek-Werner
...

\[ \Gamma \models A \]
Two main semantics for intuitionistic logic:
Two main semantics for intuitionistic logic:

- Heyting algebras [Lipton, Okada]
Two main semantics for intuitionistic logic:

- Heyting algebras [Lipton, Okada]
- Kripke structures
Two main semantics for intuitionistic logic:

- **Kripke structures**

A Kripke Structure (KS) is a tuple $\langle K, \leq, D, \models \rangle$:
A semantic for deduction modulo

Two main semantics for intuitionistic logic:

- Kripke structures

A Kripke Structure (KS) is a tuple \( \langle K, \leq, D, \Vdash \rangle \):

- \( K \) the set of worlds, partially ordered with \( \leq \) (a “temporal relation”: past, present, possible futures: partial information)
A semantic for deduction modulo

Two main semantics for intuitionistic logic:

- Kripke structures

A Kripke Structure (KS) is a tuple $\langle K, \leq, D, \models \rangle$:

- $K$ the set of worlds, partially ordered with $\leq$ (a “temporal relation”: past, present, possible futures: partial information)
- $D : \alpha \rightarrow Set$ a monotone function (interpretation domain for terms).
Two main semantics for intuitionistic logic:

- **Kripke structures**

A Kripke Structure (KS) is a tuple \( \langle K, \leq, D, \vdash \rangle \):

- \( K \) the set of worlds, partially ordered with \( \leq \) (a “temporal relation”: past, present, possible futures: partial information)
- \( D : \alpha \rightarrow \text{Set} \) a monotone function (interpretation domain for terms).
- \( \vdash \) is a relation between worlds and formulas, verifying:

A semantic for deduction modulo

- $P$ atomic: if $\alpha \leq \beta$ and $\alpha \models P$, then $\beta \models P$.
- $\alpha \models A \Rightarrow B$ iff for any $\beta \geq \alpha$, when $\beta \models A$ then $\beta \models B$.
- $\alpha \models A \lor B$ iff $\alpha \models A$ or $\alpha \models B$. 
A semantic for deduction modulo

- $P$ atomic: if $\alpha \leq \beta$ and $\alpha \vdash P$, then $\beta \vdash P$.
- $\alpha \vdash A \Rightarrow B$ iff for any $\beta \geq \alpha$, when $\beta \vdash A$ then $\beta \vdash B$.
- $\alpha \vdash A \lor B$ iff $\alpha \vdash A$ or $\alpha \vdash B$.
- Additional constraint in deduction modulo:

\[
A \equiv B \quad \text{implies} \quad \alpha \vdash A \iff \alpha \vdash B
\]
Kripke structures at work

- $A \lor (\neg A)$ is well-known not to be valid in intuitionistic logic.
- We build a structure that is invalidating this formula. Note: at least two worlds (single world = boolean model).
- $\neg A = A \Rightarrow \bot$

\[ \begin{align*}
\beta \vdash A \\
\Downarrow \\
\alpha \vdash \emptyset
\end{align*} \]
Kripke structures at work

- \( A \lor (\neg A) \) is well-known not to be valid in intuitionistic logic.
- We build a structure that is invalidating this formula. Note: at least two worlds (single world = boolean model).
- \( \neg A = A \Rightarrow \bot \)

\[
\begin{align*}
\beta & \vdash A \\
\alpha & \vdash \emptyset & \beta & \vdash A \\
\alpha & \vdash \emptyset & \alpha & \nvdash A, \neg A, A \lor \neg A
\end{align*}
\]
Constructive proof: the algorithm behind

$\Gamma \vdash A$  \quad \text{soundness} \quad \Rightarrow \quad \Gamma \models A$

Gentzen
Tait-Girard
Dowek-Werner
...

$\Gamma \vdash_{cf} A$  \quad \text{strong completeness} 

$\Gamma \models A$
Constructive proof: the algorithm behind

\[ \Gamma \vdash A \quad \text{soundness} \quad \Gamma \models A \]

\[ \Gamma 
\vdash_{cf} A \quad \text{strong completeness} \quad \text{tableaux soundness} \quad \text{tableaux completeness} \]

\[ \text{Tab} (T\emptyset \vdash \Gamma, F\emptyset \vdash A) \leftarrow \odot \]
Constructive proof: the algorithm behind

\[ \Gamma \vdash \Delta \quad \text{soundness} \quad \Gamma \models A \]

\[ \Gamma \vdash_{cf} \Delta \quad \text{strong completeness} \]

\[ \text{Tab} \left( T\emptyset \models \Gamma, \quad F\emptyset \models \Delta \right) \leftrightarrow \oslash \quad \text{tableaux completeness} \]
The tableau method

- Searching for a counter-model
The tableau method

- Searching for a counter-model
- Exhaustive algorithm, each *branch* represents a possible counter-model.
The tableau method

- Searching for a counter-model
- Exhaustive algorithm, each *branch* represents a possible counter-model.
- some rules:

\[
\begin{align*}
Tp ⊩ A ∨ B & \quad Fp ⊩ A ∨ B \\
& \quad | \quad Fp ⊩ A \\
& \quad | \quad Fp ⊩ B \\
Tp ⊩ A & \quad \quad Fp ⊩ A → B \\
& \quad | \quad Tq ⊩ A \\
& \quad | \quad Fq ⊩ B \\
Tp ⊩ B & \quad Tq ⊩ A \\
& \quad | \quad Fq ⊩ B \\
\end{align*}
\]

with proviso on q
The tableau method

- Searching for a counter-model
- Exhaustive algorithm, each *branch* represents a possible counter-model.
- some rules:

  \[ \begin{align*}
  Tp & \models A \lor B \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Tp & \models A \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Tp & \models B \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Fp & \models A \lor B \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Fp & \models A \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Fp & \models B \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Fp & \models A \Rightarrow B \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Tq & \models A \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Tq & \models B \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Fq & \models A \\
  & \quad \text{branch}
  \end{align*} \]

  \[ \begin{align*}
  Fq & \models B \\
  & \quad \text{branch}
  \end{align*} \]

  with proviso on *q*

- in deduction modulo: allow rewrite rules, define a new systematic research algorithm with *R*. 
We want to show “$A \lor B \vdash C \Rightarrow A$”

Translation in tableau language: there is NO (node of no) Kripke structure satisfying $A \lor B$ without satisfying also $C \Rightarrow A$. Let’s see if the counter-model search fails or not.

We choose as usual sequences of integers for the set of worlds (partial order: prefix).

$T\emptyset \models A \lor B, F\emptyset \not\models C \Rightarrow A$
Tableau: example 1

\( T\emptyset \vdash A \lor B, \ F\emptyset \vdash C \Rightarrow A \)
Tableau: example 1

\[
T\emptyset \models A \lor B, F\emptyset \models C \Rightarrow A
\]
\[
\quad T1 \models C
\]
\[
\quad F1 \models A
\]
Tableau: example 1

\[ T\emptyset \vdash A \lor B, F\emptyset \vdash C \Rightarrow A \]

\[ T1 \vdash C \]

\[ F1 \vdash A \]
Tableau: example 1

\[ T\emptyset \vDash A \lor B, F\emptyset \vDash C \implies A \]

\[ T1 \vDash C \]

\[ F1 \vDash A \]

\[ T\emptyset \vDash A \quad T\emptyset \vDash B \]
Tableau: example 1

$T\emptyset \vdash A \lor B, F\emptyset \vdash C \Rightarrow A$

$T1 \vdash C$

$F1 \vdash A$

$T\emptyset \vdash A \quad T\emptyset \vdash B$
Tableau: example 1

\[ T\emptyset \models A \lor B, F\emptyset \models C \Rightarrow A \]

\[ T1 \models C \]

\[ F1 \models A \]

\[ T\emptyset \models A \quad T\emptyset \models B \]
We want to show \( \vdash (A \implies B) \implies (A \implies B) \)

\[
F_\emptyset \models (A \implies B) \implies A \implies B
\]
Tableau: example 2

\[
F_\emptyset \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B
\]

\[
T_1 \vdash (A \Rightarrow B)
\]

\[
F_1 \vdash A \Rightarrow B
\]
Tableau: example 2

\[
F_{\emptyset} \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B
\]

\[
T_1 \vdash (A \Rightarrow B)
\]

\[
F_1 \vdash A \Rightarrow B
\]

\[
F_1 \vdash A \quad T_1 \vdash B
\]
Tableau: example 2

\[
F_\emptyset \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B
\]

\[
T_1 \vdash (A \Rightarrow B)
\]

\[
F_1 \vdash A \Rightarrow B
\]

\[
F_1 \vdash A \quad T_1 \vdash B
\]

\[
T_1 \vdash (A \Rightarrow B)
\]
Tableau: example 2

\[ F_\emptyset \models (A \implies B) \implies A \implies B \]

\[ T_1 \models (A \implies B) \]

\[ F_1 \models A \implies B \]

\[ T_1 \models A \]

\[ T_1 \models (A \implies B) \]

\[ T_{11} \models A \]

\[ F_{11} \models B \]
Tableau: example 2

\( F_\emptyset \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B \)

\( T_1 \vdash (A \Rightarrow B) \)

\( F_1 \vdash A \Rightarrow B \)

\( T_1 \vdash B \)

\( F_1 \vdash A \)

\( T_1 \vdash A \Rightarrow B \)

\( T_{11} \vdash A \)

\( F_{11} \vdash B \)

\( F_{11} \vdash A \)

\( T_{11} \vdash B \)
Tableau: example 2

\[ F_\emptyset \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B \]

\[ T_1 \vdash (A \Rightarrow B) \]

\[ F_1 \vdash A \Rightarrow B \]

\[ F_1 \vdash A \quad T_1 \vdash B \]

\[ T_1 \vdash (A \Rightarrow B) \]

\[ T_{11} \vdash A \]

\[ F_{11} \vdash B \]

\[ F_{11} \vdash A \quad T_{11} \vdash B \]

\[ \odot \quad \odot \]
Tableau: example 2

\[ F_\emptyset \models (A \Rightarrow B) \Rightarrow A \Rightarrow B \]

\[ T_1 \models (A \Rightarrow B) \]

\[ F_1 \models A \Rightarrow B \]

\[ F_1 \models A \quad T_1 \models B \]

\[ T_1 \models (A \Rightarrow B) \]

\[ T_{11} \models A \]

\[ F_{11} \models B \]

\[ F_{11} \models A \quad T_{11} \models B \]

\[ \emptyset \]
Tableaux completeness

- If the systematic tableau generation fails (does not terminate): does it generate a counter-model?
- Well known in the classical sequent calculus.
Tableaux completeness

- If the systematic tableau generation fails (does not terminate): does it generate a counter-model?
- Well known in the classical sequent calculus.
  - Defining a model from an infinite branch: the latter has the needed properties.
Tableaux completeness

- If the systematic tableau generation fails (does not terminate): does it generate a counter-model?
- well known in the classical sequent calculus.
  - defining a model from an infinite branch: the latter has the needed properties.
  - the model is consistent with the branch:

\[ Tp \models P \quad \text{iff} \quad p \models P \]
Tableaux completeness

- If the systematic tableau generation fails (does not terminate): does it generate a counter-model?
- well known in the classical sequent calculus.
  - defining a model from an infinite branch: the latter has the needed properties.
  - the model is consistent with the branch:
    \[ Tp \vDash P \iff p \vDash P \]
- deduction modulo: it has also to be a model of the rewrite rules \( R \).
Tableaux completeness

- If the systematic tableau generation fails (does not terminate): does it generate a counter-model?
- well known in the classical sequent calculus.
  - defining a model from an infinite branch: the latter has the needed properties.
  - the model is consistent with the branch:

\[ Tp \not\models P \iff p \not\models P \]

- deduction modulo: it has also to be a model of the rewrite rules \( R \).
- constructive point of view: if there is no counter-model, does the method terminate? (KS definition is modified)
Remember the tableau for $A \lor B \vdash C \Rightarrow A$:

- $T\emptyset \vdash A \lor B$, $F\emptyset \vdash C \Rightarrow A$

  - $T1 \vdash C$
  - $F1 \vdash A$
    - $T\emptyset \vdash A$
    - $T\emptyset \vdash B$

- the right path generates counter model.
- the nerve: the atomic formulas each world entails (forces), extension by induction.
Conditions on rewrite rules

Providing the confluence of the rewrite system \( \mathcal{R} \), and for:

- an order condition: \( > \), well-founded, having the subformula property, and such that \( P \rightarrow^* Q \) implies \( P > Q \).

the tableau method is complete.
Conditions on rewrite rules

Providing the confluence of the rewrite system $\mathcal{R}$, and for:

- an order condition: $\succ$, well-founded, having the subformula property, and such that $P \rightarrow^* Q$ implies $P \succ Q$.

- a positivity condition: if $A \rightarrow P$ then $P$ has only positive occurrences of atoms.

the tableau method is complete.
Conditions on rewrite rules

Providing the confluence of the rewrite system $\mathcal{R}$, and for:

- an order condition: $\succ$, well-founded, having the subformula property, and such that $P \rightarrow^* Q$ implies $P \succ Q$.
- a positivity condition: if $A \rightarrow P$ then $P$ has only positive occurrences of atoms.
- both conditions mixed: $\mathcal{R}_\succ \cup \mathcal{R}_+$, with a compatibility condition.

the tableau method is complete.
Conditions on rewrite rules

Providing the confluence of the rewrite system $\mathcal{R}$, and for:

- an order condition: $\succ$, well-founded, having the subformula property, and such that $P \rightarrow^* Q$ implies $P \succ Q$.
- a positivity condition: if $A \rightarrow P$ then $P$ has only positive occurrences of atoms.
- both conditions mixed: $\mathcal{R}_{\succ} \cup \mathcal{R}_+$, with a compatibility condition.
- the rule:

$$R \in R \rightarrow \forall y \left( \forall x (y \in x \Rightarrow R \in x) \Rightarrow (y \in R \Rightarrow (A \Rightarrow A))) \right)$$

the tableau method is complete.
\[ \Gamma \vdash \Delta \quad \text{soundness} \quad \Gamma \models \Delta \]

\[ \Gamma \vdash_{cf} \Delta \]

\[ \text{Tab} \left( T\emptyset \vdash \Gamma, \quad F\emptyset \vdash \Delta \right) \hookrightarrow \odot \]

\[ \text{tableaux completeness} \]
Tableaux soundness

We show the following theorem:

**Theorem**

*If a tableau starting with \( T\emptyset \vdash \Gamma, F\emptyset \vdash P \) is closed, then we can transform it into a proof of \( \Gamma \vdash_{\text{cf}} P \).*

▶ intuitionistic difficulty: in a tableau, there might be more than one “non true” formula:

\[
\begin{align*}
F\emptyset & \vdash P \lor Q \\
\quad & \vdash \\
F\emptyset & \vdash P \\
\quad & \vdash \\
F\emptyset & \vdash Q
\end{align*}
\]
Tableaux soundness

We show the following theorem:

**Theorem**

If a tableau starting with $T\emptyset \vdash \Gamma, F\emptyset \vdash P$ is closed, then we can transform it into a proof of $\Gamma \vdash_{cf} P$.

- intuitionistic difficulty: in a tableau, there might be more than one “non true” formula:

  $F\emptyset \vdash P \lor Q$

  $F\emptyset \vdash P$

  $F\emptyset \vdash Q$

- we must derive the following rule:

  $\Gamma \vdash_{cf} A \lor B \quad \Gamma \vdash_{cf} A \lor C$

  $\Gamma \vdash_{cf} A \lor (B \land C)$
we must derive the following rule:

\[
\frac{\Gamma \vdash_{cf} A \lor B \quad \Gamma \vdash_{cf} A \lor C}{\Gamma \vdash_{cf} A \lor (B \land C)}
\]
we must derive the following rule:

\[ \Gamma \vdash_{cf} A \lor B \quad \Gamma \vdash_{cf} A \lor C \]

\[ \Gamma \vdash_{cf} A \lor (B \land C) \]

easy with the cut rule:

\[
\begin{array}{c}
\Gamma, A \lor B, A \lor C \vdash A \lor (B \land C) \\
\text{Cut} \\
\Gamma, A \lor B \vdash A \lor C \\
\text{Hyp.} \\
\Gamma \vdash A \lor B \\
\text{Hyp.}
\end{array}
\]

\[
\begin{array}{c}
\Gamma, A \lor B \vdash A \lor (B \land C) \\
\text{Cut} \\
\Gamma \vdash A \lor (B \land C) \\
\end{array}
\]

\[
\begin{array}{c}
\Gamma, A \lor B \vdash A \lor C \\
\text{Hyp.}
\end{array}
\]

\[
\begin{array}{c}
\Gamma \vdash A \lor B \\
\text{Hyp.}
\end{array}
\]
we must derive the following rule:

\[
\frac{\Gamma \vdash_{cf} A \lor B \quad \Gamma \vdash_{cf} A \lor C}{\Gamma \vdash_{cf} A \lor (B \land C)}
\]

easy with the cut rule:

\[
\begin{array}{c}
\text{Hyp.} \\
\Gamma, A \lor B, A \lor C \vdash A \lor (B \land C) \\
\text{Cut} \\
\Gamma, A \lor B \vdash A \lor (B \land C) \\
\text{Hyp.} \\
\Gamma, A \lor B \vdash A \lor C \\
\text{Cut} \\
\Gamma, A \lor B \vdash A \lor (B \land C) \\
\text{Hyp.} \\
\Gamma \vdash A \lor B \\
\end{array}
\]

Without the cut rule, we show the lemma (by a double induction):

\[
\begin{array}{c}
\Gamma_1 \vdash_{cf} A \lor B \\
\Gamma_2 \vdash_{cf} A \lor C \\
\text{then} \\
\Gamma_1, \Gamma_2 \vdash_{cf} A \lor (B \land C)
\end{array}
\]
Computational content: what kind of algorithm?

Let’s reconsider the rule:

\[ R \in R \to \forall y (\forall x (y \in x \Rightarrow R \in x) \Rightarrow (y \in R \Rightarrow (A \Rightarrow A))) \]

- has semantical cut elimination but no normalization.
Let's reconsider the rule:

\[ R \in R \rightarrow \forall y (\forall x (y \in x \Rightarrow R \in x) \Rightarrow (y \in R \Rightarrow (A \Rightarrow A))) \]

- has semantical cut elimination but no normalization.
- this can not be a normalization algorithm.
Computational content: what kind of algorithm?

Let's reconsider the rule:

\[ R \in R \rightarrow \forall y \left( \forall x (y \in x \Rightarrow R \in x) \Rightarrow (y \in R \Rightarrow (A \Rightarrow A)) \right) \]

- has semantical cut elimination but no normalization.
- this can not be a normalization algorithm.
- it is more or less the tableau method described here.
This diagram does not commute.
This diagram does not commute.

But: normalization methods “generate” a certain kind of semantical cut elimination proof [Dowek - Hermant].