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: $\wedge, \vee, \Rightarrow, \neg$, and quantifiers $\forall, \exists$.

$$
\begin{gathered}
Even(0) \\
\forall n(Even(n) \Rightarrow Odd(n+1)) \\
\forall n(Odd(n) \Rightarrow Even(n+1))
\end{gathered}
$$

Deduction systems: the logical framework

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

$$\begin{gathered} \mathit{Even}(0) \\ \forall n(\mathit{Even}(n) \Rightarrow \mathit{Odd}(n+1)) \\ \forall n(\mathit{Odd}(n) \Rightarrow \mathit{Even}(n+1)) \end{gathered}$$

- a sequent :

$$\overbrace{\Gamma}^{\text{hyp.}} \vdash \overbrace{A}^{\text{conc.}}$$

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

Deduction System : sequents calculus (LJ)

- A deduction rule:

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}$$

- right and left rules

$$\frac{}{\Gamma, A \vdash A}\text{axiom} \qquad \frac{\Gamma, A \vdash B \quad \Gamma \vdash A}{\Gamma \vdash B}\text{cut}$$

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\wedge\text{-r} \qquad \frac{\Gamma, A, B \vdash C}{\Gamma, A \wedge B \vdash C}\wedge\text{-l}$$

$$\frac{\Gamma, \forall x A[x], A[t] \vdash B}{\Gamma, \forall x A[x] \vdash B}\forall\text{-g, any } t \qquad \frac{\Gamma \vdash A[x]}{\Gamma \vdash \forall x A[x]}\forall\text{-r}, x \text{ free}$$

Example: 1

$$\forall x P(x) \vdash P(0) \land P(1)$$

Example: 1

$$\dfrac{\forall x P(x) \vdash P(0) \qquad \forall x P(x) \vdash P(1)}{\forall x P(x) \vdash P(0) \wedge P(1)}\ \wedge\text{-r}$$

Example: 1

$$\dfrac{\forall\text{-l}\ \dfrac{\forall x P(x), P(0) \vdash P(0)}{\forall x P(x) \vdash P(0)} \qquad \dfrac{\forall x P(x), P(1) \vdash P(1)}{\forall x P(x) \vdash P(1)}\ \forall\text{-l}}{\forall x P(x) \vdash P(0) \land P(1)}\ \land\text{-r}$$

Example: 1

$$\dfrac{\forall\text{-l}\,\dfrac{\dfrac{}{\forall x P(x), P(0) \vdash P(0)}\,\text{axiom}}{\forall x P(x) \vdash P(0)} \qquad \dfrac{\dfrac{}{\forall x P(x), P(1) \vdash P(0)}\,\text{axiom}}{\forall x P(x) \vdash P(1)}\,\forall\text{-l}}{\forall x P(x) \vdash P(0) \wedge P(1)}\,\wedge\text{-r}$$

Example: 2

$$\forall x P(x) \vdash P(0) \land P(1)$$

Example: 2

$$\dfrac{\dfrac{\forall x P(x), P(1), P(0) \vdash P(0) \wedge P(1)}{\forall x P(x), P(0) \vdash P(0) \wedge P(1)}\ \forall\text{-l}}{\forall x P(x) \vdash P(0) \wedge P(1)}\ \forall\text{-l}$$

Example: 2

$$\dfrac{\dfrac{\dfrac{}{\forall x P(x), P(1), P(0) \vdash P(0)}\ \text{axiom} \qquad \dfrac{}{\forall x P(x), P(1), P(0) \vdash P(1)}\ \text{axiom}}{\dfrac{\forall x P(x), P(1), P(0) \vdash P(0) \wedge P(1)}{\forall x P(x), P(0) \vdash P(0) \wedge P(1)}\ \forall\text{-l}}\ \wedge\text{-r}}{\forall x P(x) \vdash P(0) \wedge P(1)}\ \forall\text{-l}$$

Example: 2

$$
\dfrac{\dfrac{\dfrac{\overline{\forall x P(x), P(1), P(0) \vdash P(0)}\ \text{axiom} \qquad \overline{\forall x P(x), P(1), P(0) \vdash P(1)}\ \text{axiom}}{\forall x P(x), P(1), P(0) \vdash P(0) \land P(1)}\ \land\text{-r}}{\forall x P(x), P(0) \vdash P(0) \land P(1)}\ \forall\text{-l}}{\forall x P(x) \vdash P(0) \land P(1)}\ \forall\text{-l}
$$

- the first rule is not always "don't care": free variable condition.

Axioms vs. rewriting

Axioms	Rewriting
$x + S(y) = S(x + y)$	$x + S(y) \rightarrow S(x + y)$
$x + 0 = x$	$x + 0 \rightarrow x$
$x * 0 = 0$	$x * 0 \rightarrow 0$
$x * S(y) = x + x * y$	$x * S(y) \rightarrow x + x * y$
$(x * y = 0) \Leftrightarrow (x = 0 \vee y = 0)$	$(x * y = 0) \rightarrow (x = 0 \vee y = 0)$
$\dfrac{\dfrac{\vdots}{\mathcal{T} \vdash 2 * 2 = 4}}{\mathcal{T} \vdash \exists x(2 * x = 4)}$	$\dfrac{\dfrac{}{\vdash 4 = 4}}{\vdash \exists x(2 * x = 4)}$

Deduction modulo: allowed rewriting

- General form (free variables are possible):

$$l \rightarrow r$$

Deduction modulo: allowed rewriting

- General form (free variables are possible):

$$l \to r$$

- use: We replace $t = \sigma l$ by σr (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 σr (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 σr (unification). Rewriting could be deep in the term.
- rewriting on terms:
$$x + S(y) \rightarrow S(x + y)$$
- and on **propositions** (predicate symbols):
$$x * y = 0 \rightarrow x = 0 \vee y = 0$$
- advantage: expressiveness

Deduction modulo: allowed rewriting

- General form (free variables are possible):

$$l \to r$$

- use: We replace $t = \sigma l$ by σr (unification). Rewriting could be deep in the term.
- rewriting on terms:

$$x + S(y) \to S(x + y)$$

- and on **propositions** (predicate symbols):

$$x * y = 0 \to x = 0 \vee y = 0$$

- advantage: expressiveness
- we obtain a congruence modulo $\mathcal{R}$ (chosen set of rules): $\equiv$

Deduction modulo: allowed rewriting

- General form (free variables are possible):

$$l \to r$$

- use: We replace $t = \sigma l$ by σr (unification). Rewriting could be deep in the term.
- rewriting on terms:

$$x + S(y) \to S(x + y)$$

- and on **propositions** (predicate symbols):

$$x * y = 0 \to x = 0 \vee y = 0$$

- advantage: expressiveness
- we obtain a congruence modulo $\mathcal{R}$ (chosen set of rules): $\equiv$
- deduction rules transform as such:

$$\text{axiom}\ \frac{}{\Gamma, A \vdash A} \qquad \text{becomes} \qquad \frac{}{\Gamma, A \vdash B}\ \text{axiom}, A \equiv B$$

Deduction modulo : sequent calculus modulo

$$\frac{}{\Gamma, A \vdash B}\text{axiom } A \equiv B \qquad \frac{\Gamma, A \vdash C \quad \Gamma \vdash B}{\Gamma \vdash C}\text{cut } A \equiv B$$

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash C} \wedge\text{-r } A \wedge B \equiv C \qquad \frac{\Gamma, A, B \vdash C}{\Gamma, D \vdash C} \wedge\text{-l } A \wedge B \equiv D$$

$$\frac{\Gamma, B, A[t] \vdash C}{\Gamma, B \vdash C}\forall\text{-l } \forall x A[x] \equiv B \qquad \frac{\Gamma \vdash A[x]}{\Gamma \vdash B}\forall\text{-r}^* \ \forall x A[x] \equiv B$$

Example: 3

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

$$\begin{array}{rcl} P(0) & \rightarrow & A \\ P(1) & \rightarrow & B \end{array}$$

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

Example: 3

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

$$
\begin{aligned}
P(0) &\quad\longrightarrow\quad A \\
P(1) &\quad\longrightarrow\quad B
\end{aligned}
$$

$$
\dfrac{\forall x P(x) \vdash A \qquad \forall x P(x) \vdash B}{\forall x P(x) \vdash A \wedge B}\ \wedge\text{-r}
$$

Example: 3

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

$$\begin{array}{rcl} P(0) & \longrightarrow & A \\ P(1) & \longrightarrow & B \end{array}$$

$$\dfrac{\forall\text{-l}\ \dfrac{\forall x P(x), P(0) \vdash A}{\forall x P(x) \vdash A} \qquad \dfrac{\forall x P(x), P(1) \vdash B}{\forall x P(x) \vdash B}\ \forall\text{-l}}{\forall x P(x) \vdash A \wedge B}\ \wedge\text{-r}$$

Example: 3

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

$$
\begin{array}{rcl}
P(0) & \rightarrow & A \\
P(1) & \rightarrow & B
\end{array}
$$

$$
\dfrac{\dfrac{\dfrac{}{\forall x P(x), P(0) \vdash B}\ \text{axiom}}{\forall x P(x) \vdash A}\ \forall\text{-r} \qquad \dfrac{\dfrac{}{\forall x P(x), P(1) \vdash B}\ \text{axiom}}{\forall x P(x) \vdash B}\ \forall\text{-r}}{\forall x P(x) \vdash A \wedge B}\ \wedge\text{-r}
$$

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 $\mathcal{R}$:

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

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

Example: 4

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

$$\begin{array}{rcl} P(0) & \to & A \\ P(1) & \to & B \end{array}$$

$$\dfrac{\forall x P(x), A \vdash A \wedge B \qquad \forall x P(x) \vdash A}{\forall x P(x) \vdash A \wedge B}\ \text{cut}$$

Example: 4

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

$$\begin{array}{rcl} P(0) & \to & A \\ P(1) & \to & B \end{array}$$

$$\dfrac{\forall x P(x), A \vdash A \wedge B \qquad \dfrac{\dfrac{Ax.}{\forall x P(x), P(0) \vdash A}}{\forall x P(x) \vdash A}\,\forall\text{-r}}{\forall x P(x) \vdash A \wedge B}\,\text{cut}$$

Example: 4

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

$$
\begin{array}{rcl}
P(0) & \rightarrow & A \\
P(1) & \rightarrow & B
\end{array}
$$

$$
\begin{prooftree}
\AxiomC{Ax.}
\UnaryInfC{$\forall x P(x), A \vdash A$}
\AxiomC{Ax.}
\UnaryInfC{$\forall x P(x), P(1), A \vdash B$}
\RightLabel{$\forall$-r}
\UnaryInfC{$\forall x P(x), A \vdash B$}
\LeftLabel{$\wedge$-r}
\BinaryInfC{$\forall x P(x), A \vdash A \wedge B$}
\AxiomC{Ax.}
\UnaryInfC{$\forall x P(x), P(0) \vdash A$}
\RightLabel{$\forall$-}
\UnaryInfC{$\forall x P(x) \vdash A$}
\RightLabel{cut}
\BinaryInfC{$\forall x P(x) \vdash A \wedge B$}
\end{prooftree}
$$

Example: 4

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

$$
\begin{array}{lcl}
P(0) & \rightarrow & A \\
P(1) & \rightarrow & B
\end{array}
$$

$$
\dfrac{
  \wedge\text{-r}\ \dfrac{
    \dfrac{\text{Ax.}}{\forall x P(x), A \vdash A}
    \qquad
    \dfrac{\dfrac{\text{Ax.}}{\forall x P(x), P(1), A \vdash B}}{\forall x P(x), A \vdash B}\ \forall\text{-r}
  }{\forall x P(x), A \vdash A \wedge B}
  \qquad
  \dfrac{\dfrac{\text{Ax.}}{\forall x P(x), P(0) \vdash A}}{\forall x P(x) \vdash A}\ \forall\text{-r}
}{\forall x P(x) \vdash A \wedge B}\ \text{cut}
$$

- an unnecessary detour
- we could have cutted on any formula!

The cut rule: a detour

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

- 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 !

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.

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.

$$\Gamma \vdash A \quad \triangleright \quad \Gamma \vdash_{cf} A$$

- 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 *do not* proceed by cuts !
- in theory: consistence, proof normalization (Curry-Howard) depend of its elimination.
- eliminating cuts: a key result.

$$\Gamma \vdash A \quad \triangleright \quad \Gamma \vdash_{cf} A$$

- two main paths towards:
  - proof normalization (syntactic).
  - semantical methods.
- in deduction modulo: indecidable, need for general criterions on $\mathcal{R}$

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

$$
\begin{tikzcd}[column sep=huge, row sep=huge]
\Gamma \vdash A \arrow[r, shift left, "\text{soundness}"] \arrow[d, "\begin{array}{c}\text{Gentzen}\\ \text{Tait-Girard}\\ \text{Dowek-Werner}\\ \ldots\end{array}"'] & \Gamma \models A \arrow[l, shift left, "\text{completeness}"] \\
\Gamma \vdash_{cf} A &
\end{tikzcd}
$$

The semantical method

$$
\begin{tikzcd}
\Gamma \vdash A \arrow[rr, shift left, "\text{soundness}"] \arrow[d, "\substack{\text{Gentzen} \\ \text{Tait-Girard} \\ \text{Dowek-Werner} \\ \dots}"'] & & \Gamma \models A \arrow[ll, shift left, "\text{completeness}"] \arrow[dll, "\text{strong completeness}"] \\
\Gamma \vdash_{cf} A & &
\end{tikzcd}
$$

The semantical method

$$
\begin{tikzcd}
\Gamma \vdash A \arrow[rr, "\text{soundness}"] \arrow[d, "\begin{array}{c}\text{Gentzen}\\ \text{Tait-Girard}\\ \text{Dowek-Werner}\\ \ldots\end{array}"'] & & \Gamma \models A \arrow[dll, "\text{strong completeness}"] \\
\Gamma \vdash_{cf} A & &
\end{tikzcd}
$$

A semantic for deduction modulo

Two main semantics for intuitionistic logic:

A semantic for deduction modulo

Two main semantics for intuitionistic logic:

- Heyting algebras [Lipton,Okada]

A semantic for deduction modulo

Two main semantics for intuitionistic logic:

- Heyting algebras [Lipton,Okada]
- Kripke structures

A semantic for deduction modulo

Two main semantics for intuitionistic logic:

- Kripke structures

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

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 $\langle K, \leq, D, \Vdash \rangle$:

- K the set of worlds, partially ordered with $\leq$ (a “temporal relation”: past, present, possible futures: partial information)
- $D : \alpha \to \mathcal{S}et$ a monotone function (interpretation domain for terms).

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)
- $D : \alpha \rightarrow \mathcal{S}et$ a monotone function (interpretation domain for terms).
- $\Vdash$ is a relation between worlds and formulas, verifiying:

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 $\beta \Vdash B$.
- $\alpha \Vdash A \vee B$ iff $\alpha \Vdash A$ or $\alpha \Vdash B$.

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 $\beta \Vdash B$.
- $\alpha \Vdash A \vee B$ iff $\alpha \Vdash A$ or $\alpha \Vdash B$.
- Additional constraint in deduction modulo:

$$A \equiv B \quad \text{implies} \quad \alpha \Vdash A \Leftrightarrow \alpha \Vdash B$$

Kripke structures at work

- $A \vee (\neg A)$ 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).
- $\neg A = A \Rightarrow \bot$

$$\begin{tikzcd}
\beta \Vdash A \arrow[d, dash] \\
\alpha \Vdash \emptyset
\end{tikzcd}$$

Kripke structures at work

- $A \vee (\neg A)$ 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).
- $\neg A = A \Rightarrow \bot$

$$\begin{tikzcd}
\beta \Vdash A \arrow[d, dash] \\
\alpha \Vdash \emptyset
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\beta \Vdash A \arrow[d, dash] \\
\alpha \Vdash \emptyset \text{ and } \alpha \nVdash A, \neg A, A \vee \neg A
\end{tikzcd}$$

Constructive proof: the algorithm behind

$$\begin{tikzcd}[column sep=huge, row sep=large]
\Gamma \vdash A \arrow[r, "\text{soundness}"] \arrow[d, swap, "\begin{array}{c}\text{Gentzen}\\ \text{Tait-Girard}\\ \text{Dowek-Werner}\\ \ldots\end{array}"] & \Gamma \models A \arrow[dl, "\text{strong completeness}"] \\
\Gamma \vdash_{cf} A &
\end{tikzcd}$$

Constructive proof: the algorithm behind

$$
\begin{tikzcd}
\Gamma \vdash A \arrow[rr, "\text{soundness}"] & & \Gamma \models A \arrow[dll, "\text{strong completeness}"'] \arrow[d, shift left, "\text{tableaux completeness}"] \\
\Gamma \vdash_{cf} A & & \mathrm{Tab}\,(T\emptyset \Vdash \Gamma,\ F\emptyset \Vdash A) \hookrightarrow \odot \arrow[u, shift left, "\text{tableaux soundness}"]
\end{tikzcd}
$$

Constructive proof: the algorithm behind

$$
\begin{tikzcd}
\Gamma \vdash \Delta \arrow[rr, "\text{soundness}"] & & \Gamma \models A \arrow[d, "\text{tableaux completeness}"] \arrow[dll, "\text{strong completeness}"'] \\
\Gamma \vdash_{cf} \Delta & & \mathrm{Tab}\,(T\emptyset \Vdash \Gamma,\ F\emptyset \Vdash \Delta) \hookrightarrow \odot \arrow[ll, "\text{tableaux soundness}"]
\end{tikzcd}
$$

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:

$$\begin{array}{c} Tp \Vdash A \vee B \\ \swarrow \qquad \searrow \\ Tp \Vdash A \qquad Tp \Vdash B \end{array} \qquad\qquad \begin{array}{c} Fp \Vdash A \vee B \\ | \\ Fp \Vdash A \\ | \\ Fp \Vdash B \end{array}$$

$$\begin{array}{c} Tp \Vdash A \Rightarrow B \\ \swarrow \qquad \searrow \\ Tq \Vdash B \qquad Fq \Vdash A \end{array} \qquad\qquad \begin{array}{c} Fp \Vdash A \Rightarrow B \\ | \\ Tq \Vdash A \\ | \\ Fq \Vdash B \end{array}$$

with proviso on q

The tableau method

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

$$\begin{array}{c} Tp \Vdash A \vee B \\ \hline Tp \Vdash A \quad \mid \quad Tp \Vdash B \end{array} \qquad\qquad \begin{array}{c} Fp \Vdash A \vee B \\ | \\ Fp \Vdash A \\ | \\ Fp \Vdash B \end{array}$$

$$\begin{array}{c} Tp \Vdash A \Rightarrow B \\ \hline Tq \Vdash B \quad \mid \quad Fq \Vdash A \end{array} \qquad\qquad \begin{array}{c} Fp \Vdash A \Rightarrow B \\ | \\ Tq \Vdash A \\ | \\ Fq \Vdash B \end{array}$$

with proviso on q

- in deduction modulo: allow rewrite rules, define a new systematic research algorithm with $\mathcal{R}$.

Tableau: example 1

- We want to show "$A \vee B \vdash C \Rightarrow A$"
- tranlsation in tableau language: there is NO (node of no) Kripke structure satisfying $A \vee B$ 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).

$T\emptyset \Vdash A \vee B, F\emptyset \Vdash C \Rightarrow A$

Tableau: example 1

$T\emptyset \Vdash A \vee B, F\emptyset \Vdash C \Rightarrow A$

Tableau: example 1

$$T\emptyset \Vdash A \vee B, F\emptyset \Vdash C \Rightarrow A$$

$$T1 \Vdash C$$

$$F1 \Vdash A$$

Tableau: example 1

$$
\begin{array}{c}
T\emptyset \Vdash A \vee B, F\emptyset \Vdash C \Rightarrow A \\
| \\
T1 \Vdash C \\
| \\
F1 \Vdash A
\end{array}
$$

Tableau: example 1

$$T\emptyset \Vdash A \vee B, F\emptyset \Vdash C \Rightarrow A$$

$$T1 \Vdash C$$

$$F1 \Vdash A$$

$$T\emptyset \Vdash A \qquad T\emptyset \Vdash B$$

Tableau: example 1

$$T\emptyset \Vdash A \vee B,\; F\emptyset \Vdash C \Rightarrow A$$

$$T1 \Vdash C$$

$$F1 \Vdash A$$

$$T\emptyset \Vdash A \qquad T\emptyset \Vdash B$$

$$\odot$$

Tableau: example 1

$$T\emptyset \Vdash A \vee B, F\emptyset \Vdash C \Rightarrow A$$

$$T1 \Vdash C$$

$$F1 \Vdash A$$

$$T\emptyset \Vdash A \qquad T\emptyset \Vdash B$$

$$\odot$$

Tableau: example 2

- We want to show "$\vdash (A \Rightarrow B) \Rightarrow (A \Rightarrow B)$"

$F_{\emptyset} \Vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B$

Tableau: example 2

$$F_{\emptyset} \Vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B$$

$$T_1 \Vdash (A \Rightarrow B)$$

$$F_1 \Vdash A \Rightarrow B$$

Tableau: example 2

$$F_{\emptyset} \Vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B$$

$$T_1 \Vdash (A \Rightarrow B)$$

$$F_1 \Vdash A \Rightarrow B$$

$$F_1 \Vdash A \qquad T_1 \Vdash B$$

Tableau: example 2

$$
\begin{tikzcd}[arrows=dash]
 & F_{\emptyset} \Vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B \arrow[d] & \\
 & T_1 \Vdash (A \Rightarrow B) \arrow[d] & \\
 & F_1 \Vdash A \Rightarrow B \arrow[dl] \arrow[dr] & \\
F_1 \Vdash A \arrow[d] & & T_1 \Vdash B \\
T_1 \Vdash (A \Rightarrow B) & &
\end{tikzcd}
$$

Tableau: example 2

$$F_{\emptyset} \Vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B$$

$$T_1 \Vdash (A \Rightarrow B)$$

$$F_1 \Vdash A \Rightarrow B$$

$$F_1 \Vdash A \qquad\qquad T_1 \Vdash B$$

$$T_1 \Vdash (A \Rightarrow B)$$

$$T_{11} \Vdash A$$

$$F_{11} \Vdash B$$

Tableau: example 2

$$F_\emptyset \Vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B$$

$$T_1 \Vdash (A \Rightarrow B)$$

$$F_1 \Vdash A \Rightarrow B$$

$$F_1 \Vdash A \qquad\qquad T_1 \Vdash B$$

$$T_1 \Vdash (A \Rightarrow B)$$

$$T_{11} \Vdash A$$

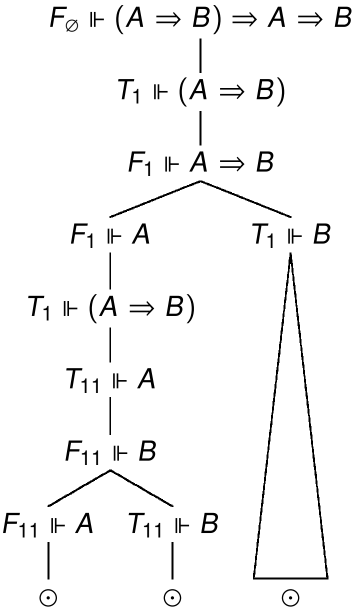
$$F_{11} \Vdash B$$

$$F_{11} \Vdash A \qquad T_{11} \Vdash B$$

Tableau: example 2

- $F_\emptyset \Vdash (A \Rightarrow B) \Rightarrow A \Rightarrow B$
  - $T_1 \Vdash (A \Rightarrow B)$
    - $F_1 \Vdash A \Rightarrow B$
      - $F_1 \Vdash A$
        - $T_1 \Vdash (A \Rightarrow B)$
          - $T_{11} \Vdash A$
            - $F_{11} \Vdash B$
              - $F_{11} \Vdash A$
                - $\odot$
              - $T_{11} \Vdash B$
                - $\odot$
      - $T_1 \Vdash B$

Tableau: example 2



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:

$$Tp \Vdash P \quad \text{iff} \quad p \Vdash P$$

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:

$$Tp \Vdash P \quad \text{iff} \quad p \Vdash P$$

  - deduction modulo: it has also to be a model of the rewrite rules $\mathcal{R}$.

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:

$$Tp \Vdash P \quad \text{iff} \quad p \Vdash P$$

  - deduction modulo: it has also to be a model of the rewrite rules $\mathcal{R}$.
- constructive point of view: if there is no counter-model, does the method terminate? (KS definition is modified)

Remember the tableau for $A \vee B \vdash C \Rightarrow A$:

$T\emptyset \Vdash A \vee B, F\emptyset \Vdash C \Rightarrow A$

$T1 \Vdash C$

$F1 \Vdash A$

$T\emptyset \Vdash A \qquad T\emptyset \Vdash B$

$\odot$

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

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 occurences 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 occurences 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 occurences of atoms.
- both conditions mixed: $\mathcal{R}_\succ \cup \mathcal{R}_+$, with a compatibility condition.
- the rule:

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

the tableau method is complete.

$$
\begin{tikzcd}[column sep=huge, row sep=large]
\Gamma \vdash \Delta \arrow[r, "\text{soundness}"] & \Gamma \models \Delta \arrow[d, "\begin{array}{c}\text{tableaux}\\ \text{completeness}\end{array}"] \\
\Gamma \vdash_{cf} \Delta & \begin{array}{c}\text{Tab}\ (T\emptyset \Vdash \Gamma,\\ F\emptyset \Vdash \Delta) \hookrightarrow \odot\end{array}
\end{tikzcd}
$$

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

$$\begin{array}{c} F\emptyset \Vdash P \vee Q \\ | \\ F\emptyset \Vdash P \\ | \\ F\emptyset \Vdash Q \end{array}$$

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 diffculty: in a tableau, there might be more than one "non true" formula:

$$\begin{array}{c} F\emptyset \Vdash P \vee Q \\ | \\ F\emptyset \Vdash P \\ | \\ F\emptyset \Vdash Q \end{array}$$

- we must derive the following rule:

$$\frac{\Gamma \vdash_{cf} A \vee B \quad \Gamma \vdash_{cf} A \vee C}{\Gamma \vdash_{cf} A \vee (B \wedge C)}$$

- we must derive the following rule:

$$\frac{\Gamma \vdash_{cf} A \vee B \quad \Gamma \vdash_{cf} A \vee C}{\Gamma \vdash_{cf} A \vee (B \wedge C)}$$

- we must derive the following rule:

$$\frac{\Gamma \vdash_{cf} A \vee B \quad \Gamma \vdash_{cf} A \vee C}{\Gamma \vdash_{cf} A \vee (B \wedge C)}$$

- easy with the cut rule:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma, A \vee B, A \vee C \vdash A \vee (B \wedge C)} \quad \dfrac{\text{Hyp.}}{\Gamma, A \vee B \vdash A \vee C}}{\Gamma, A \vee B \vdash A \vee (B \wedge C)}\,\text{Cut} \quad \dfrac{\text{Hyp.}}{\Gamma \vdash A \vee B}}{\Gamma \vdash A \vee (B \wedge C)}\,\text{Cut}$$

- we must derive the following rule:

$$\frac{\Gamma \vdash_{cf} A \vee B \quad \Gamma \vdash_{cf} A \vee C}{\Gamma \vdash_{cf} A \vee (B \wedge C)}$$

- easy with the cut rule:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma, A \vee B, A \vee C \vdash A \vee (B \wedge C)} \quad \dfrac{\text{Hyp.}}{\Gamma, A \vee B \vdash A \vee C}}{\Gamma, A \vee B \vdash A \vee (B \wedge C)}\text{Cut} \quad \dfrac{\text{Hyp.}}{\Gamma \vdash A \vee B}}{\Gamma \vdash A \vee (B \wedge C)}\text{Cut}$$

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

$$\begin{array}{rl} \Gamma_1 \vdash_{cf} A \vee B & \Gamma_2 \vdash_{cf} A \vee C \\ \text{then} & \Gamma_1, \Gamma_2 \vdash_{cf} A \vee (B \wedge C) \end{array}$$

Computational content: what kind of algorithm ?

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.

Computational content: what kind of algorithm ?

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:

$$R \in R \to \forall y\ (\forall x(y \in x \Rightarrow R \in x) \Rightarrow (y \in R \Rightarrow (A \Rightarrow A)))$$

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

$$
\begin{tikzcd}[column sep=huge, row sep=huge]
\Gamma \vdash A \arrow[r, "\text{soundness}"] \arrow[d, "\begin{array}{c}\text{Gentzen}\\ \text{Tait-Girard}\\ \text{Dowek-Werner}\\ \dots\end{array}"'] & \Gamma \models A \arrow[d, "\begin{array}{c}\text{tableaux}\\ \text{soundness}\end{array}"] \\
\Gamma \vdash_{cf} A & \begin{array}{c}\text{Tab}\,(T\emptyset \Vdash \Gamma,\\ F\emptyset \Vdash A) \hookrightarrow \odot\end{array} \arrow[l, "\text{tableaux soundness}"]
\end{tikzcd}
$$

- This diagram does not commute.

$$\begin{tikzcd}
\Gamma \vdash A \arrow[rr, "\text{soundness}"] \arrow[d, "\text{Gentzen}\\ \text{Tait-Girard}\\ \text{Dowek-Werner}\\ \ldots"'] & & \Gamma \models A \arrow[d, "\text{tableaux}\\ \text{soundness}"] \\
\Gamma \vdash_{cf} A & & \mathrm{Tab}\,(T\emptyset \Vdash \Gamma,\ F\emptyset \Vdash A) \hookrightarrow \odot \arrow[ll, "\text{tableaux soundness}"']
\end{tikzcd}$$

- This diagram does not commute.
- But: normalization methods “generate” a certain kind of semantical cut elimination proof [Dowek - Hermant].