A Linear Logic Modulo

Olivier Hermant

June 29, 2007
• Linear Logic has much to say about connectors.
• Deduction Modulo has much to say about (first-order) quantifiers.
Linear Logic has much to say about connectors.
Deduction Modulo has much to say about (first-order) quantifiers.
let’s combine them.
The language

- Usual first-order logic language.
- Logical connectors

\[ \otimes, \mathcal{V}, \sim, \& , \oplus , \! , ? \]

- Logical constants

\[ 1, \bot , \top , 0 \]

- First-order quantifiers \( \forall, \exists \)
The language

- Usual first-order logic language.
- Logical connectors

\[
\begin{align*}
\text{multiplicatives} & \quad \text{additives} \quad \text{exponentials} \\
\otimes, \otimes, \neg, & \quad \& , \oplus, & \quad !, ?,
\end{align*}
\]

- Logical constants

\[
\begin{align*}
\text{multiplicatives} & \quad \text{additives} \\
1, \bot & \quad \top, 0
\end{align*}
\]

- First-order quantifiers \( \forall, \exists \)

- The negation symbol \( \bot \) is not a primitive symbol
- Atoms \( A \) and negated atoms \( A^\bot \)
- We work with negation normal forms (classical LL, one sided sequent calculus)
Dualities in Linear Logic

\[ A^\perp \perp = (A^\perp)^\perp = A \]

**Multiplicatives**

\[ \bot^\perp = 1 \quad 1^\perp = \bot \]

\[ (A \otimes B)^\perp = A^\perp \otimes B^\perp \quad (A \nabla B)^\perp = A^\perp \nabla B^\perp \]

\[ A \multimap B = A^\perp \nabla B \]

**Additives**

\[ \top^\perp = 0 \quad 0^\perp = \top \]

\[ (A \oplus B)^\perp = A^\perp \& B^\perp \quad (A \& B)^\perp = A^\perp \oplus B^\perp \]

**Exponentials**

\[ (!A)^\perp = ?(A^\perp) \quad (?A)^\perp = !(A^\perp) \]

**Quantifiers**

\[ (\forall xA)^\perp = \exists xA^\perp \quad (\exists xA)^\perp = \forall xA^\perp \]
Deduction rules

- sequent style
- one-sided (duality): \( \Gamma \vdash \Delta \) is written \( \vdash \Gamma \bot, \Delta \) (negation NF)
- axiom looks like \( \vdash A \bot, A \)
Deduction rules

- sequent style
- one-sided (duality): $\Gamma \vdash \Delta$ is written $\vdash \Gamma \perp, \Delta$ (negation NF)
- axiom looks like $\vdash A \perp, A$
- independent groups of connectors (substructural logics)
- multiplicatives separate the context (perfect world)
- additives do not (imperfect world)
- contexts: sets (no permutation needed)
### Deduction rules of Linear Logic

#### Identities

<table>
<thead>
<tr>
<th>Rule</th>
<th>Syntactic Form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>init</strong></td>
<td>$\vdash A \perp, A$</td>
</tr>
</tbody>
</table>

#### Multiplicatives

<table>
<thead>
<tr>
<th>Rule</th>
<th>Syntactic Form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1-r</strong></td>
<td>$\vdash 1$</td>
</tr>
<tr>
<td><strong>⊗-r</strong></td>
<td>$\vdash A, \Gamma \vdash B, \Delta \vdash A \otimes B, \Gamma, \Delta$</td>
</tr>
</tbody>
</table>

#### Additives

<table>
<thead>
<tr>
<th>Rule</th>
<th>Syntactic Form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>no 0-r</strong></td>
<td>No rule exists</td>
</tr>
<tr>
<td><strong>&amp;-r</strong></td>
<td>$\vdash A, \Delta \vdash B, \Delta \vdash A &amp; B, \Delta$</td>
</tr>
</tbody>
</table>

#### Quantifiers

<table>
<thead>
<tr>
<th>Rule</th>
<th>Syntactic Form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>∀-r</strong></td>
<td>$\vdash A, \Delta \vdash \forall x A, \Delta$</td>
</tr>
<tr>
<td><strong>∃-r</strong></td>
<td>$\vdash (t/x) A, \Delta \vdash \exists x A, \Delta$</td>
</tr>
</tbody>
</table>

#### Exponentials

<table>
<thead>
<tr>
<th>Rule</th>
<th>Syntactic Form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>contraction</strong></td>
<td>$\vdash ?A, ?A, \Delta \vdash ?A, \Delta \vdash ?A, \Delta$</td>
</tr>
<tr>
<td><strong>weakening</strong></td>
<td>$\vdash ?A, \Delta \vdash \Delta, ?A$</td>
</tr>
<tr>
<td><strong>cut</strong></td>
<td>$\vdash A \perp, \Gamma \vdash A, \Delta \vdash \Gamma, \Delta$</td>
</tr>
<tr>
<td><strong>⊥-r</strong></td>
<td>$\vdash \bot, \Delta$</td>
</tr>
<tr>
<td><strong>⊗-r</strong></td>
<td>$\vdash A, B, \Delta \vdash A \otimes B, \Delta$</td>
</tr>
<tr>
<td><strong>&amp;-r</strong></td>
<td>$\vdash A, \Delta \vdash A &amp; B, \Delta$</td>
</tr>
<tr>
<td><strong>⊕-r1</strong></td>
<td>$\vdash B, \Delta \vdash A \oplus B, \Delta$</td>
</tr>
<tr>
<td><strong>⊕-r2</strong></td>
<td>$\vdash A \oplus B, \Delta$</td>
</tr>
<tr>
<td><strong>∀-r</strong></td>
<td>$\vdash A, \Delta \vdash \forall x A, \Delta$</td>
</tr>
<tr>
<td><strong>∃-r</strong></td>
<td>$\vdash (t/x) A, \Delta \vdash \exists x A, \Delta$</td>
</tr>
<tr>
<td><strong>dereliction</strong></td>
<td>$\vdash \Delta, ?A \vdash ?A, \Delta$</td>
</tr>
<tr>
<td><strong>promotion</strong></td>
<td>$\vdash ?A, \Delta \vdash \Delta, ?A \vdash ?A, !A$</td>
</tr>
</tbody>
</table>
Adding rewrite rules

- rewrite rules are of the two following forms:
  - on terms
    
    \[
    \begin{align*}
    x \times 0 & \rightarrow 0 \\
    x + 0 & \rightarrow 0
    \end{align*}
    \]
  
  - on propositions
    
    \[P(0) \rightarrow \forall x P(x)\]

- a set of rewrite rules \(\mathcal{R}\) defines a congruence \(\equiv\)
Adding rewrite rules

- rewrite rules are of the two following forms:
  - on terms
    \[
    x \cdot 0 \rightarrow 0 \\
    x + 0 \rightarrow 0
    \]
  - on propositions
    \[
    P(0) \rightarrow \forall x P(x)
    \]
- a set of rewrite rules \( \mathcal{R} \) defines a congruence \( \equiv \)
- it is taken into account in the rules (side condition):
  - axiom \( \vdash A \perp, A \) turns into \( \vdash B, A \) axiom, \( B \equiv A \perp \)
Adding rewrite rules

- rewrite rules are of the two following forms:
  - on terms
    
    \[
    \begin{align*}
    x \ast 0 & \rightarrow 0 \\
    x + 0 & \rightarrow 0
    \end{align*}
    \]
  - on propositions
    
    \[P(0) \rightarrow \forall x P(x)\]

- a set of rewrite rules \(R\) defines a congruence \(\equiv\)

- it is taken into account in the rules (side condition):
  
  \[
  \frac{\vdash A \perp, A}{\vdash B, A} \quad \text{axiom, } B \equiv A \perp
  \]

- many interesting examples, e.g. Church’s simple types theory: first-order encoding of higher-order LL by rewrite rules.
Rules of Linear Logic modulo

- **Identities**
  - \( \vdash A, \Gamma \vdash B, \Delta \) cut, \( A \equiv B \)

- **Multiplicatives**
  - \( \vdash \Delta \) ⊥-r, \( A \equiv \perp \)
  - \( \vdash A, \Delta \vdash B, \Delta \) \&-r, \( C \equiv A \& B \)
  - \( \vdash A, \Delta \vdash B, \Delta \) ⊗-r, \( C \equiv A \otimes B \)

- **Additives**
  - \( \vdash A, \Gamma \vdash B, \Delta \) ⊤-r, \( A \equiv \top \)
  - \( \vdash A, \Delta \vdash B, \Delta \) ⊕-r1, \( C \equiv A \oplus B \)
  - \( \vdash A, \Delta \vdash B, \Delta \) ⊕-r2, \( C \equiv A \oplus B \)

- **Quantifiers**
  - \( \vdash A, \Gamma \vdash B, \Delta \) ∀-r, \( C \equiv \forall x A, x \) fresh
  - \( \vdash (t/x)A, \Delta \vdash C, \Delta \) ∃-r, \( C \equiv \exists x A, t \) term

- **Exponentials**
  - \( \vdash \Delta, A \) derel., \( B \equiv ?A \)
  - \( \vdash \Delta, A \vdash B \) promo., \( B \equiv !A, \Delta \equiv ?\Gamma \)
A toy example

- Rewrite system:

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

- Proof of \( \vdash \exists x (P(x) \perp) \), \( A \otimes B \) (two sided: \( \forall x P(x) \vdash A \otimes B \))
A toy example

- Rewrite system:
  
  \[ P(0) \rightarrow A \]
  \[ P(1) \rightarrow B \]

- Proof of \( \vdash \exists x(P(x) \perp), A \otimes B \) (two sided: \( \forall x P(x) \vdash A \otimes B \))

  \[
  \begin{array}{c}
  \exists\text{-r} \\
  \text{dereliction}
  \\
  \exists\text{-r}
  \end{array}
  \]

  \[
  \begin{array}{c}
  \vdash P(0) \perp, A \\
  \vdash \exists x(P(x) \perp), A \\
  \vdash ? \exists x(P(x) \perp), A \\
  \vdash ? \exists x(P(x) \perp), A \otimes B \\
  \vdash ? \exists x(P(x) \perp), A \otimes B \\
  \end{array}
  \]

  \[
  \begin{array}{c}
  \vdash P(1) \perp, A \\
  \vdash \exists x(P(x) \perp), B \\
  \vdash ? \exists x(P(x) \perp), B \\
  \vdash ? \exists x(P(x) \perp), A \otimes B \\
  \vdash ? \exists x(P(x) \perp), A \otimes B \\
  \end{array}
  \]

  \[
  \begin{array}{c}
  \exists\text{-r} \\
  \text{dereliction}
  \end{array}
  \]

  \[
  \begin{array}{c}
  \exists\text{-r}
  \end{array}
  \]

  \[
  \begin{array}{c}
  \otimes\text{-r}
  \end{array}
  \]

  \[
  \begin{array}{c}
  \text{contraction}
  \end{array}
  \]
Studying cut elimination

- theoretic power of DM: in some cases, no cut elimination.
Studying cut elimination

- theoretic power of DM: in some cases, no cut elimination.
- counterexample

\[ A \rightarrow (!A) \rightarrow A \]

...can type every (untyped) \( \lambda \)-term (especially \( \Omega = \lambda x.(xx) \))
Studying cut elimination

➤ theoretic power of DM: in some cases, no cut elimination.
➤ counterexample

\[ A \to (\neg A) \to A \]

can type every (untyped) \( \lambda \)-term (especially \( \Omega = \lambda x.(xx) \))

➤ worse: this rule admits cuts but no normalization
➤ we give semantic ways to prove cut elimination (admissibility)
Phase spaces

- a topological interpretation
- idea behind: sets of contexts (i.e. $A^* = \{\Gamma \mid \Gamma \vdash A \text{ provable} \}$)
- like Boolean algebras, Heyting algebras (pseudo-complement: think about open sets!). “Natural” interpretation:

\[(A \land B)^* = A^* \cap B^*\]

intended meaning:

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

- a topological interpretation
- idea behind: sets of contexts (i.e. \( A^* = \{ \Gamma \mid \Gamma \vdash A \text{ provable} \} \))
- like Boolean algebras, Heyting algebras (pseudo-complement: think about open sets!). “Natural” interpretation:
  \[
  (A \land B)^* = A^* \cap B^*
  \]
  intended meaning:
  \[
  \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \land B}
  \]
- in LL: two conjunctions \( \otimes \) and \( \& \) : which one is the intersection?
Phase spaces

- a topological interpretation
- idea behind: sets of contexts (i.e. \( A^* = \{ \Gamma \mid \Gamma \vdash A \text{ provable} \} \))
- like Boolean algebras, Heyting algebras (pseudo-complement: think about open sets !). “Natural” interpretation:

\[(A \land B)^* = A^* \cap B^*\]

intended meaning:

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

- in LL: two conjunctions \( \otimes \) and \( \& \): which one is the intersection ?
- Hint: look at the previous rule. But what for the other ?
Phase spaces

- \((M, .)\): a commutative monoid, 1: unit, \(\perp\): a fixed subset of \(M\)
  (intended meaning: contexts with concatenation, empty context and some fixed subset – the pole)
Phase spaces

- $(M, .)$: a commutative monoid, $1$: unit, $\perp$: a fixed subset of $M$
  (intended meaning: contexts with concatenation, empty context and
  some fixed subset – the pole)
- plus special treatment for exponentials (modalities): set $J$ ...
- basic construct: orthogonal of subsets $\alpha \subseteq M$

\[
\alpha^\perp = \{ a \mid \alpha.a \subseteq \perp \}
\]
- consider only sets closed by bi-orthogonality ($\alpha = \alpha^{\perp\perp}$): facts.
  (involutive closure operator: $(_\perp)^{\perp\perp}$)
Phase spaces

- $(M, .)$: a commutative monoid, $1$: unit, $\perp$: a fixed subset of $M$
  (intended meaning: contexts with concatenation, empty context and some fixed subset – the pole)
- plus special treatment for exponentials (modalities): set $J$ ...
- basic construct: orthogonal of subsets $\alpha \subseteq M$

\[
\alpha^\perp = \{a \mid \alpha.a \subseteq \perp\}
\]

- consider only sets closed by bi-orthogonality ($\alpha = \alpha^{\perp\perp}$): facts.
  (involutive closure operator: $(\perp)^{\perp\perp}$)
- semantic operators
  - $\top = M$
  - $0 = \top^\perp = \{a \mid M.a \subseteq \perp\}$
  - $\alpha \& \beta = \alpha \cap \beta$
  - $\alpha \otimes \beta = (\alpha.\beta)^{\perp\perp}$
Phase models

- defining a model: usual business
  - base interpretation for terms and predicates
  - connectors as operators
  - quantifiers: ∀ infinite intersection (on domain), ∃ closure of infinite union

- specific condition on models. Rewrite rules valid:

  \[ A \equiv B \text{ should imply } A^* = B^* \]
Phase models

- defining a model: usual business
  - base interpretation for terms and predicates
  - connectors as operators
  - quantifiers: $\forall$ infinite intersection (on domain), $\exists$ closure of infinite union
- specific condition on models. Rewrite rules valid:

  $$A \equiv B \text{ should imply } A^* = B^*$$

- soundness holds (well ... confluence of rewrite rules required)

  $$\Gamma \vdash A \text{ implies } \Gamma^* \leq A^* \quad \text{(one sided version: } \Gamma^{\perp} \subseteq A^*)$$

- completeness also ...
Phase models for cut elimination

- ... but we can do more!

Find a model such that $\Gamma^* \leq A^*$ implies $\vdash_{cf} A, \Delta$

- Okada’s work extended to deduction modulo settings
Context phase spaces

▶ monoid $M$: set of finite contexts, composition law $\cdot$: concatenation.
▶ define the

\[
(\text{outer value}) \quad \llbracket A \rrbracket = \{ \Gamma \mid \vdash_{cf} \Gamma, A \}
\]

▶ take $\llbracket \bot \rrbracket$ for (the semantical) $\bot$. Exercise: $\{A\}^{\bot} = \llbracket A \rrbracket$
Context phase spaces

- define the

  \[(\text{outer value}) \quad \lbrack A \rbrack = \{ \Gamma \mid \vdash_{cf} \Gamma, A \}\]

- take $\lbrack \bot \rbrack$ for (the semantical) $\bot$. Exercise: $\{A\}^{\bot} = \lbrack A \rbrack$
- interpret each \textit{atomic} predicate symbol $P$ by $\lbrack P \rbrack$.
- this defines a phase space. (would also define Heyting or Boolean algebra)
Context phase spaces

- define the

\[
\text{\textbf{(outer value)}} \quad \llbracket A \rrbracket = \{ \Gamma \mid \vdash_{cf} \Gamma, A \}\]

- take $\llbracket \bot \rrbracket$ for (the semantical) $\bot$. Exercise: $\{A\}^{\bot} = \llbracket A \rrbracket$
- interpret each atomic predicate symbol $P$ by $\llbracket P \rrbracket$.
- this defines a phase space. (would also define Heyting or Boolean algebra)
- aim: $\Gamma \in \llbracket A \rrbracket$. 
Semantic cut elimination

- show $\Gamma \in \llbracket A \rrbracket$ in a few steps
- Main Lemma: for any $A$, 

$$ A^\perp \in A^* \subseteq \llbracket A \rrbracket $$
semantic cut elimination

- show $\Gamma \in \llbracket A \rrbracket$ in a few steps
- Main Lemma: for any $A$,

$$A \perp \in A^* \subseteq \llbracket A \rrbracket$$

- consequence:
  - $\Gamma^* \subseteq \llbracket \Gamma \rrbracket = \{\Gamma\}^\perp$
  - $\{\Gamma\}^{\perp\perp} \subseteq \Gamma^* \perp$ (negating the previous)
  - $\Gamma \in \{\Gamma\}^{\perp\perp}$ (exercise)
  - $\Gamma^* \perp \subseteq A^*$ (soundness)
  - $A^* \subseteq \llbracket A \rrbracket$
  - Q.E.D: $\vdash^{cf} \Gamma, A$

Additional constraint:

$A^* = B^*$ when $A \equiv B$

dependent on $\equiv$

we do that for two conditions on rewrite rules: order and positivity. Plus a combination of both.
show $\Gamma \in \llbracket A \rrbracket$ in a few steps

Main Lemma: for any $A$,

$$A^\bot \in A^* \subseteq \llbracket A \rrbracket$$

consequence:

$\Gamma^* \subseteq \llbracket \Gamma \rrbracket = \{\Gamma\}^\bot$

$\{\Gamma\}^{\bot\bot} \subseteq \Gamma^*^{\bot}$ (negating the previous)

$\Gamma \in \{\Gamma\}^{\bot\bot}$ (exercise)

$\Gamma^*^{\bot} \subseteq A^*$ (soundness)

$A^* \subseteq \llbracket A \rrbracket$

Q.E.D: $\vdash_{cf} \Gamma, A$

Stop! Additional constraint: $A^* = B^*$ when $A \equiv B$

dependent on $\equiv$

we do that for two conditions on rewrite rules: order and positivity. Plus a combination of both.
The positivity condition in short
Core ideas
- define proof nets for linear logic modulo
- study the proof normalization algorithms
- define some pseudo-Phase spaces (as Truth values algebras)