

Set Comprehension in Church's Type Theory

Automated Theorem Proving in Extensional Type Theory

Chad E. Brown

`cebrown@andrew.cmu.edu`

Department of Mathematical Sciences

Carnegie Mellon University

Simply Typed λ -calculus

Simple Types \mathcal{T} :

o (truth values)
 ι (individuals)
 $(\alpha\beta)$ (functions from β to α)

Simply Typed λ -calculus

Simple Types \mathcal{T} :

o (truth values)
 ι (individuals)
 $(\alpha\beta)$ (functions from β to α)

Terms:

x_α Variables (\mathcal{V})
 A_α Parameters (\mathcal{P})
 c_α Logical Constants (\mathcal{S})
 $[\mathbf{F}_{\alpha\beta} \mathbf{B}_\beta]_\alpha$ Application
 $[\lambda y_\beta \mathbf{A}_\alpha]_{\alpha\beta}$ λ -abstraction

Propositions

Propositions:

$$\begin{aligned} & A_0 \\ & [A_\alpha \stackrel{\alpha}{=} B_\alpha] \\ & \top \\ & \neg M \\ & [M \vee N] \\ & [\forall x_\alpha M] \end{aligned}$$

Propositions

Propositions:

$$\begin{aligned} & A_0 \\ [A_\alpha \stackrel{\alpha}{=} B_\alpha] \\ \top \\ \neg M \\ [M \vee N] \\ [\forall x_\alpha M] \end{aligned}$$

Abbreviations:

$[M \supset N]$	means	$[\neg M \vee N]$
$[M \wedge N]$	means	$\neg[\neg M \vee \neg N]$
$[M \equiv N]$	means	$[[M \supset N] \wedge [N \supset M]]$
$[\exists x_\alpha M]$	means	$\neg[\forall x_\alpha \neg M]$

Propositions

Propositions:

$$\begin{aligned} & A_0 \\ & [A_\alpha \overset{\alpha}{=} B_\alpha] \\ & \top \\ & \neg M \\ & [M \vee N] \\ & [\forall x_\alpha M] \end{aligned}$$

Sentences = Closed Propositions

Automated Theorem Proving

- Given: M – Goal: Find a Proof of M

Automated Theorem Proving

- Given: M – Goal: Find a Proof of M
- TPS - Higher-Order Theorem Proving System (Andrews)

Automated Theorem Proving

- Given: M – Goal: Find a Proof of M
- TPS - Higher-Order Theorem Proving System (Andrews)
- Traditional TPS Search Procedures:

Automated Theorem Proving

- Given: M – Goal: Find a Proof of M
- TPS - Higher-Order Theorem Proving System (Andrews)
- Traditional TPS Search Procedures:
 - Use Expansion Trees

Automated Theorem Proving

- Given: M – Goal: Find a Proof of M
- TPS - Higher-Order Theorem Proving System (Andrews)
- Traditional TPS Search Procedures:
 - Use Expansion Trees
 - Matings (Andrews, Bibel)

Automated Theorem Proving

- Given: M – Goal: Find a Proof of M
- TPS - Higher-Order Theorem Proving System (Andrews)
- Traditional TPS Search Procedures:
 - Use Expansion Trees
 - Matings (Andrews, Bibel)
 - Higher-Order Unification (Huet)

Automated Theorem Proving

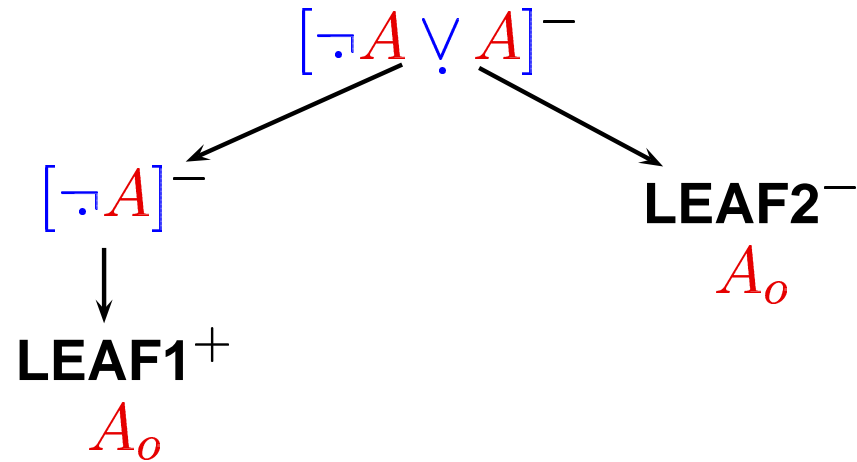
- Given: M – Goal: Find a Proof of M
- TPS - Higher-Order Theorem Proving System (Andrews)
- Traditional TPS Search Procedures:
 - Use Expansion Trees
 - Matings (Andrews, Bibel)
 - Higher-Order Unification (Huet)
 - Primitive Substitutions (Andrews)

Automated Theorem Proving

- Given: M – Goal: Find a Proof of M
- TPS - Higher-Order Theorem Proving System (Andrews)
- Traditional TPS Search Procedures:
 - Use Expansion Trees
 - Matings (Andrews, Bibel)
 - Higher-Order Unification (Huet)
 - Primitive Substitutions (Andrews)
- Searches in Elementary Type Theory
(No Extensionality)

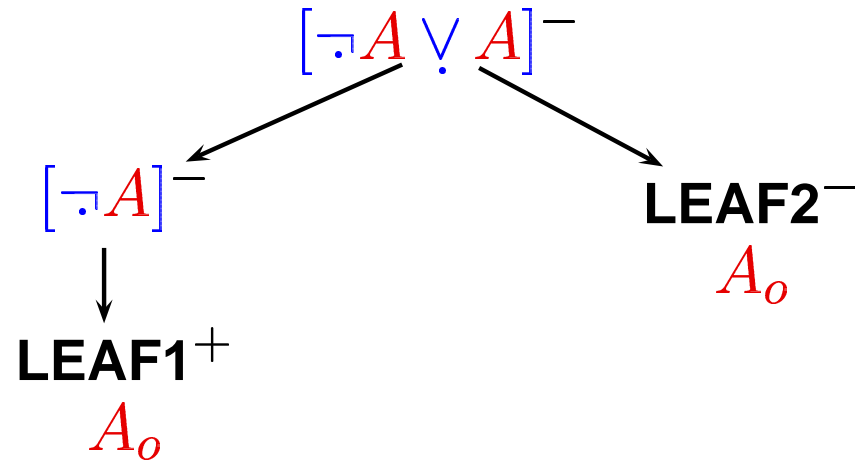
Expansion Trees: Simple Example

Expansion Tree for $[\neg A \vee A]$:



Expansion Trees: Simple Example

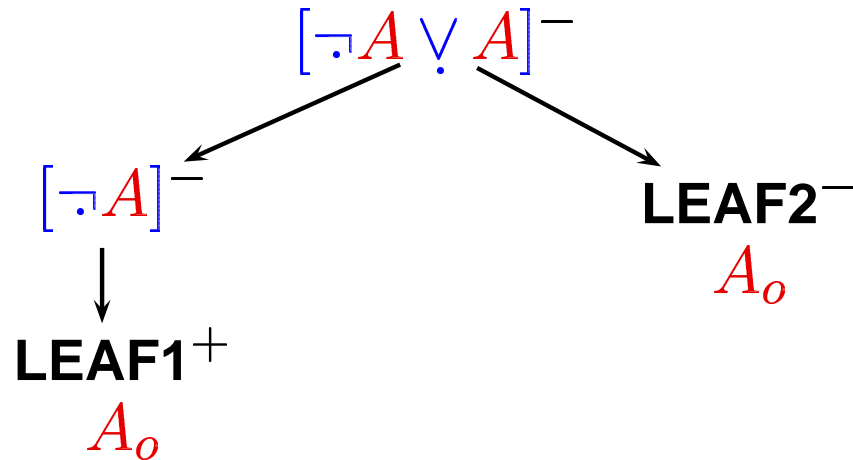
Expansion Tree for $[\neg A \vee A]$:



Connection: (**LEAF1** . **LEAF2**)

Expansion Trees: Simple Example

Expansion Tree for $[\neg A \vee A]$:

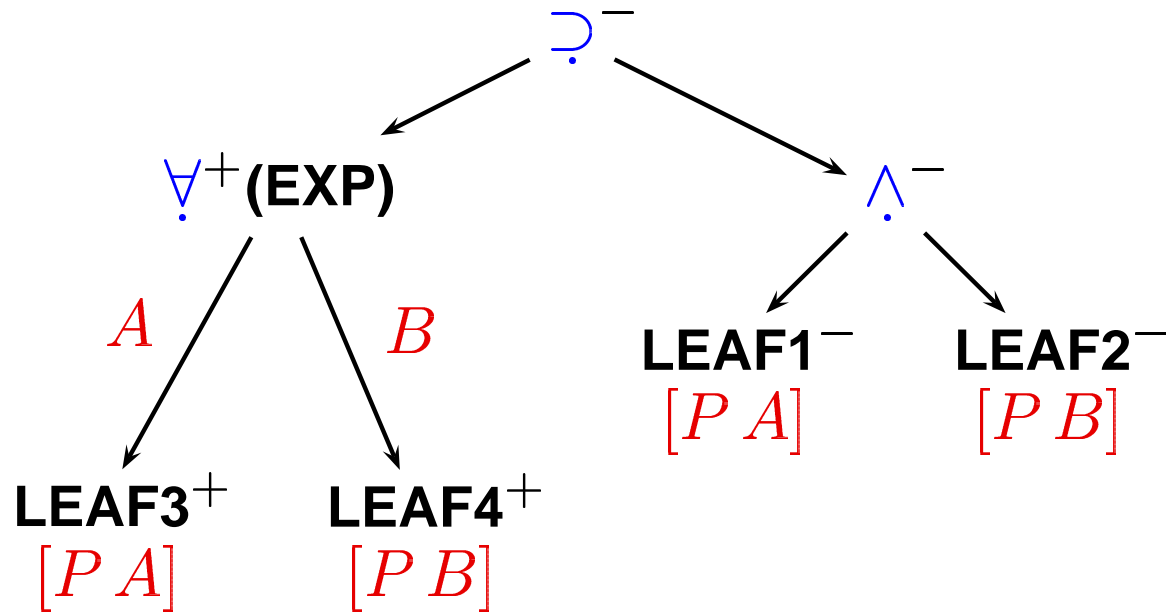


Connection: (LEAF1 . LEAF2)

Expansion Proof: Expansion Tree + Complete Mating

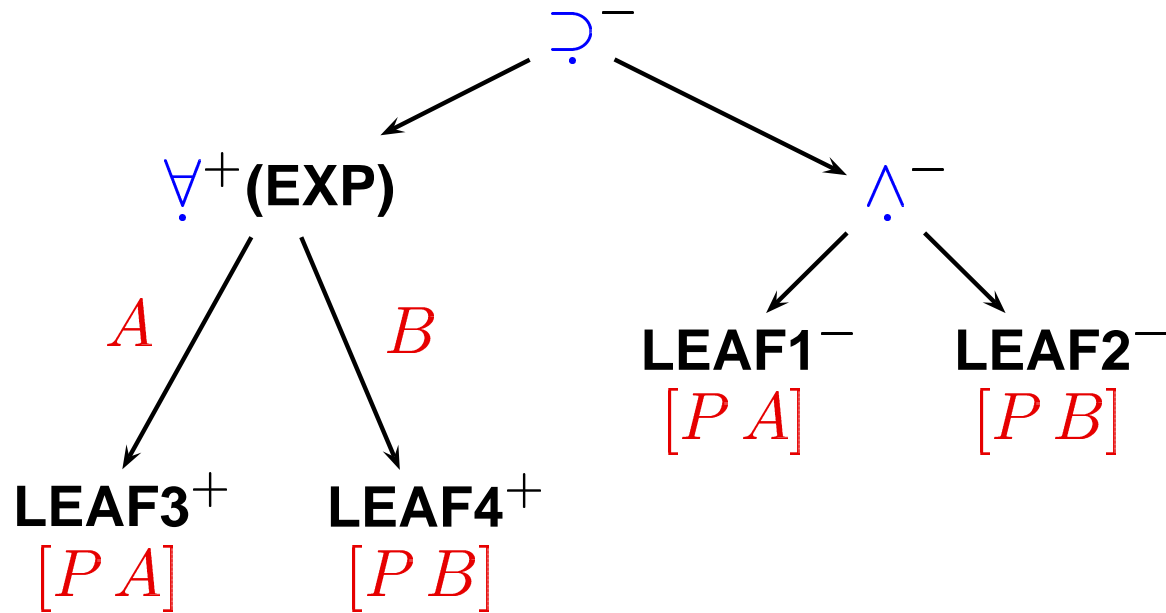
Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \lrcorner [P A_t] \wedge [P B_t]]$:

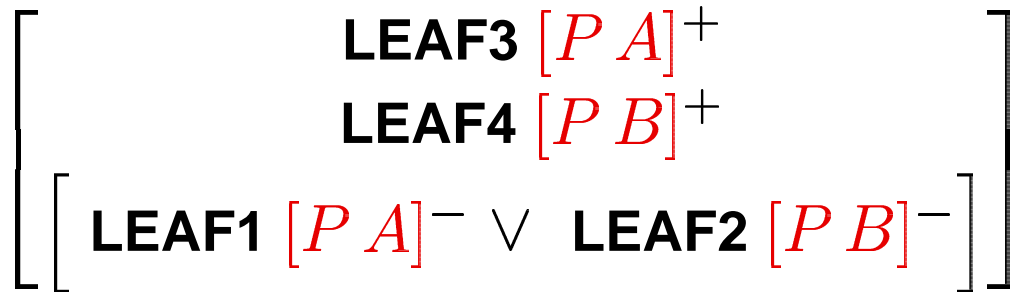


Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

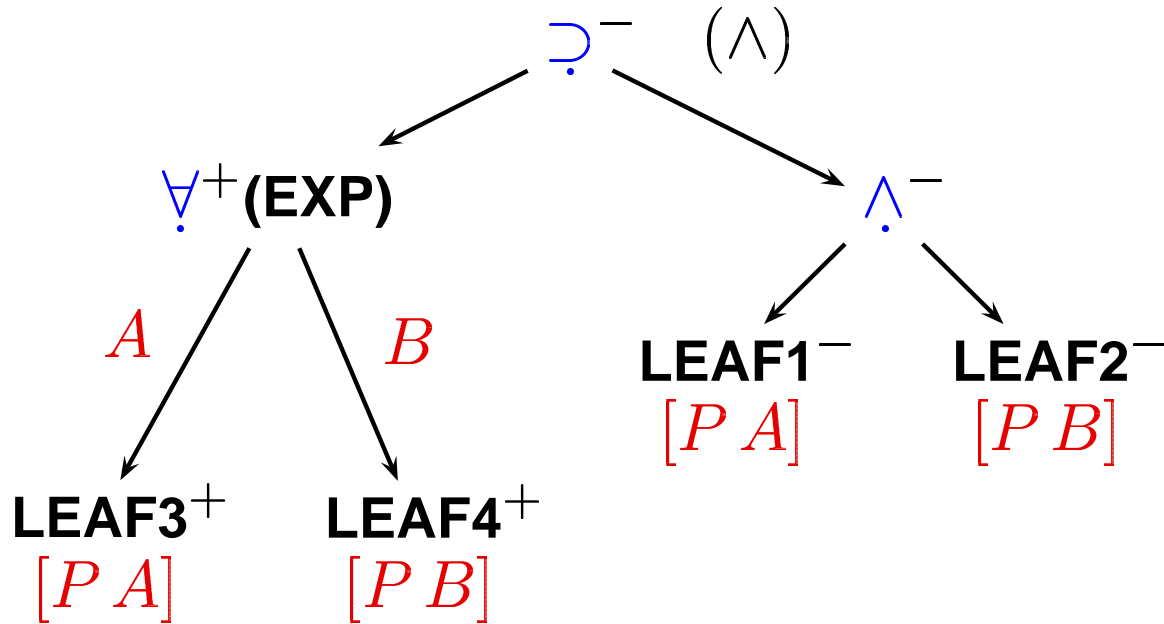


JForm:

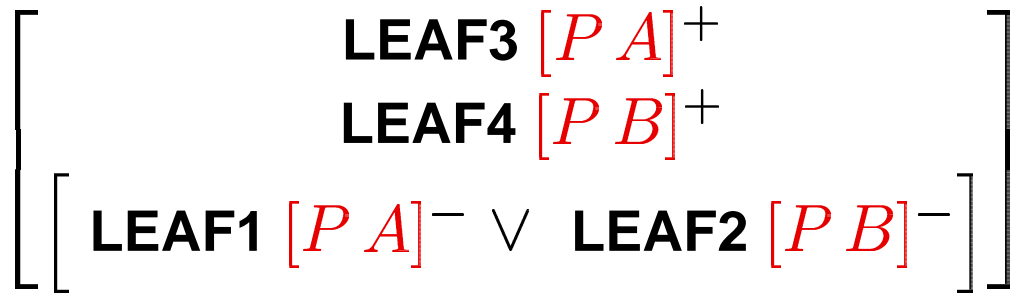


Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

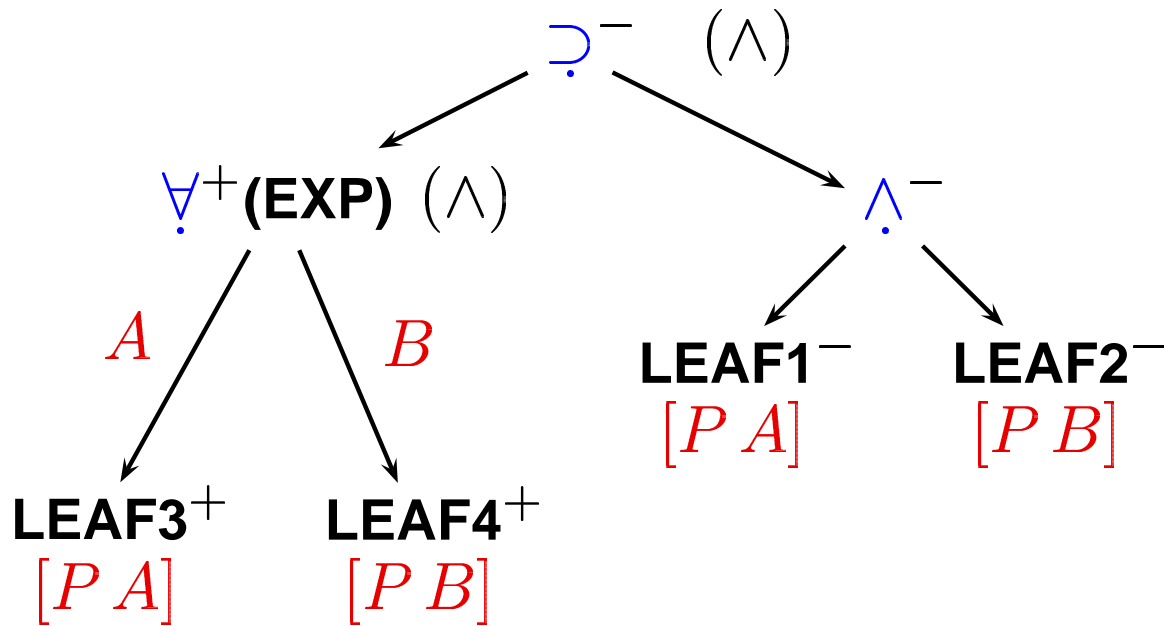


Vertical Paths
= Conjunctive Paths

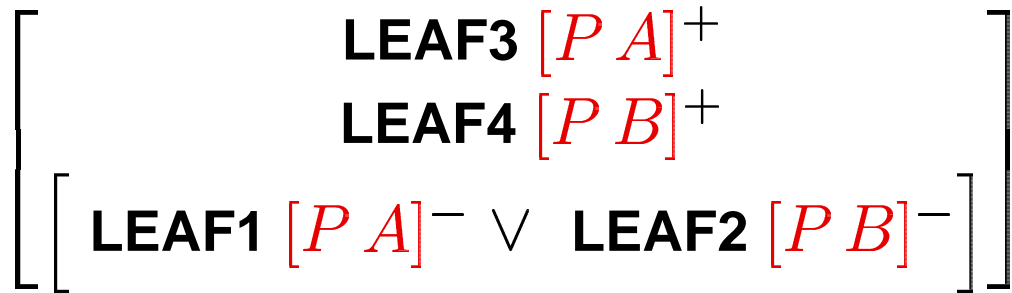


Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

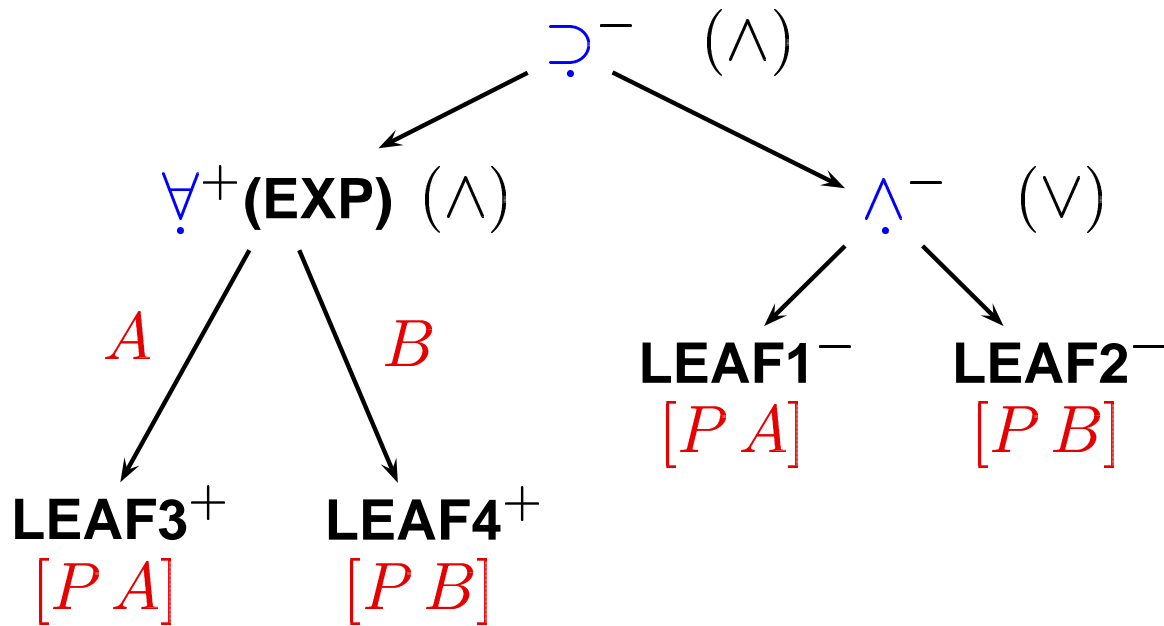


Vertical Paths
= Conjunctive Paths

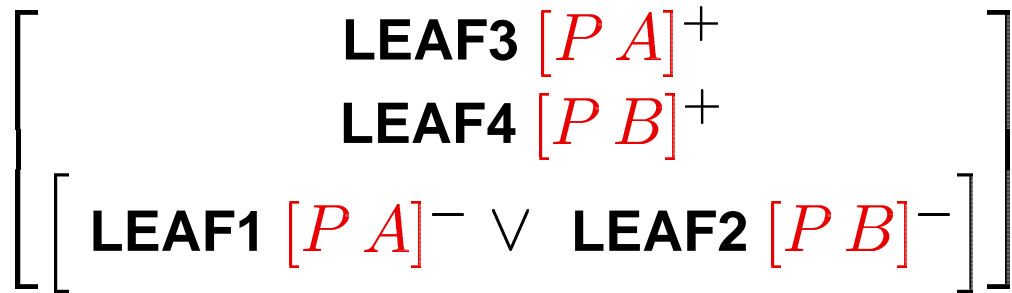


Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

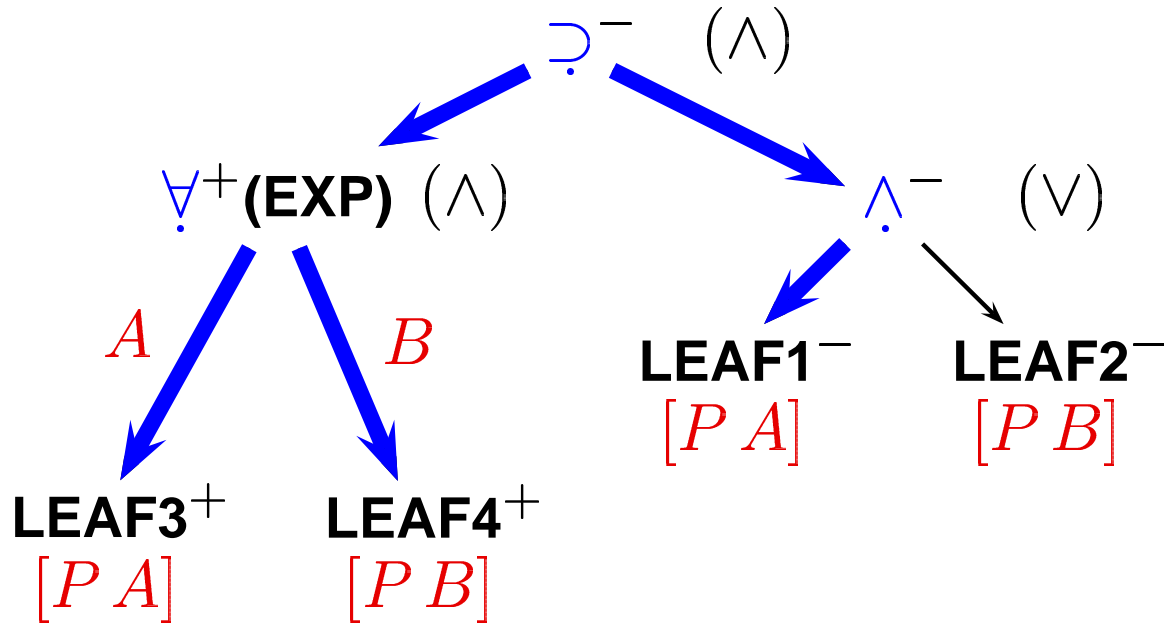


Vertical Paths
= Conjunctive Paths

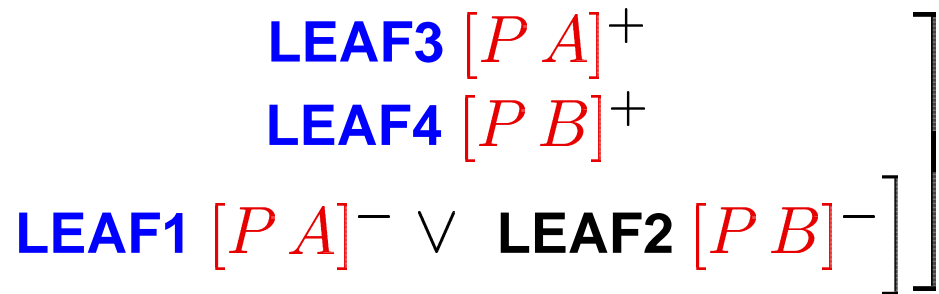


Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \lrcorner [P A_t] \wedge [P B_t]]$:

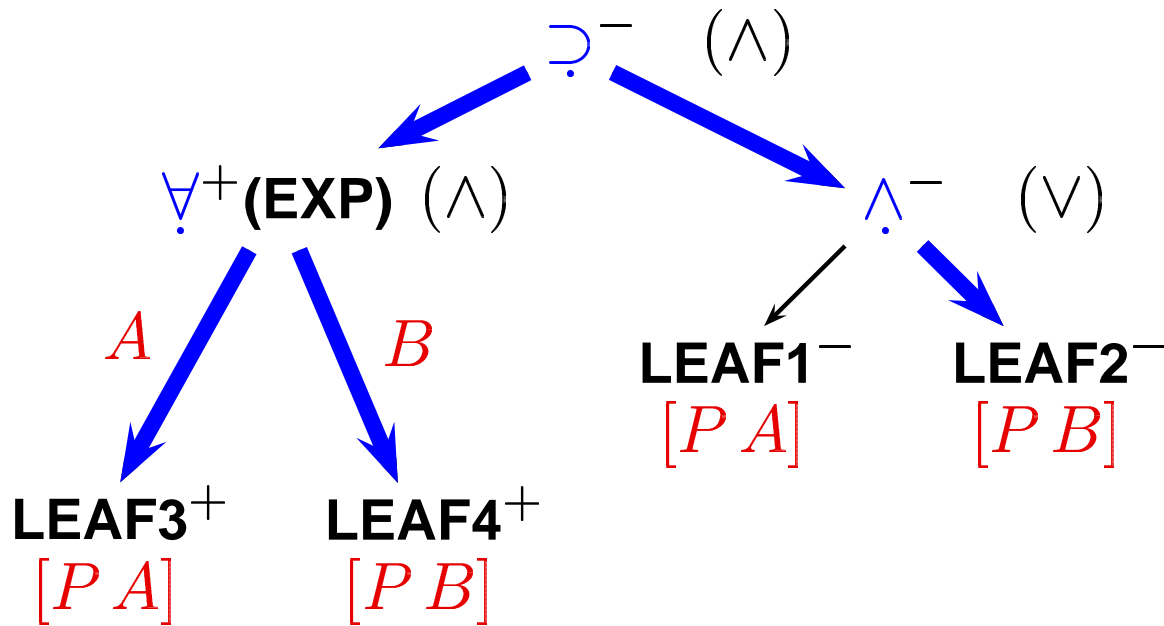


Vertical Paths
= Conjunctive Paths

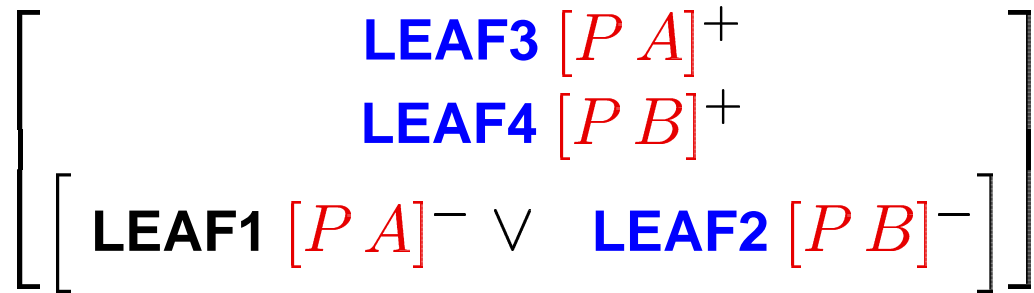


Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \lrcorner [P A_t] \wedge [P B_t]]$:

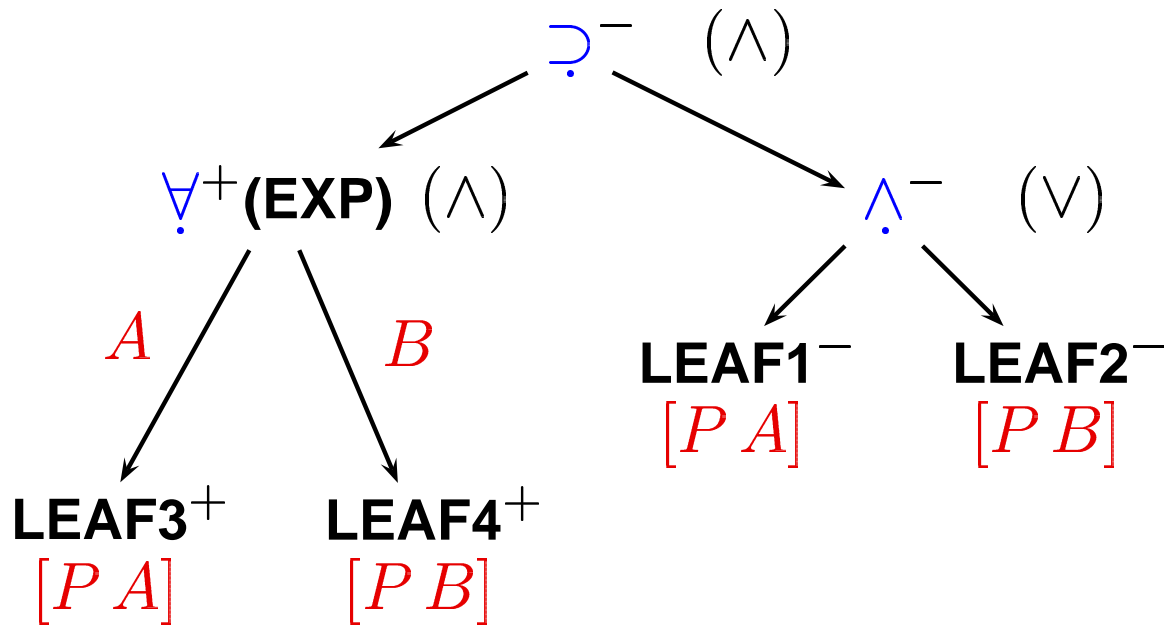


Vertical Paths
= Conjunctive Paths



Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

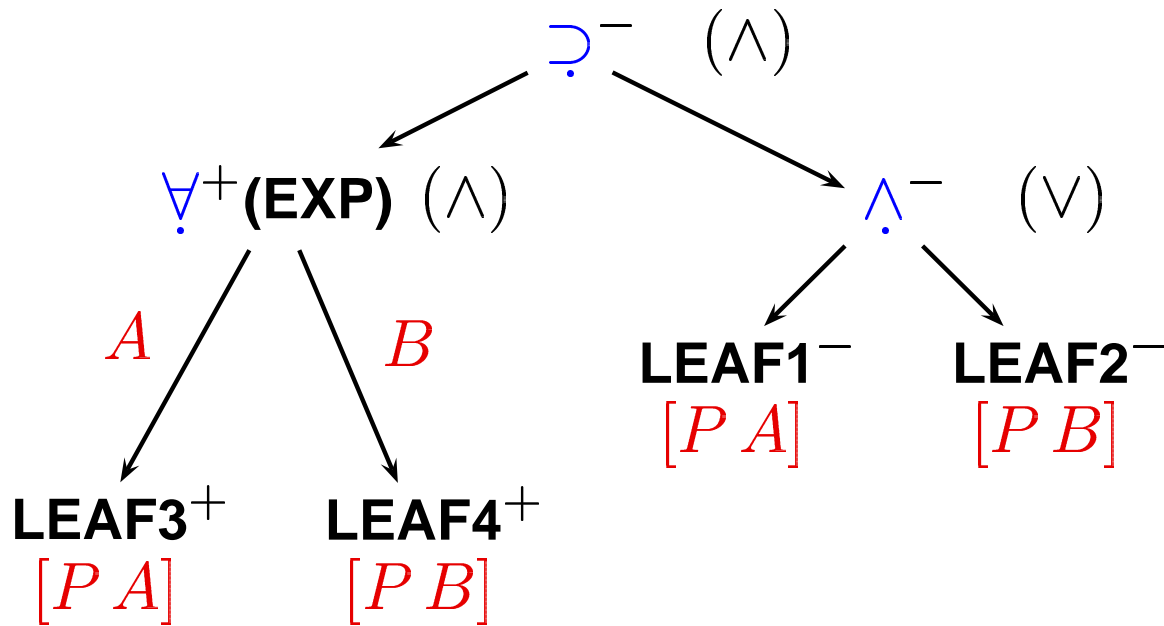


Complete Mating: $\left[\begin{array}{c} \text{LEAF3 } [P A]^+ \\ \text{LEAF4 } [P B]^+ \\ \left[\text{LEAF1 } [P A]^- \vee \text{LEAF2 } [P B]^- \right] \end{array} \right]$

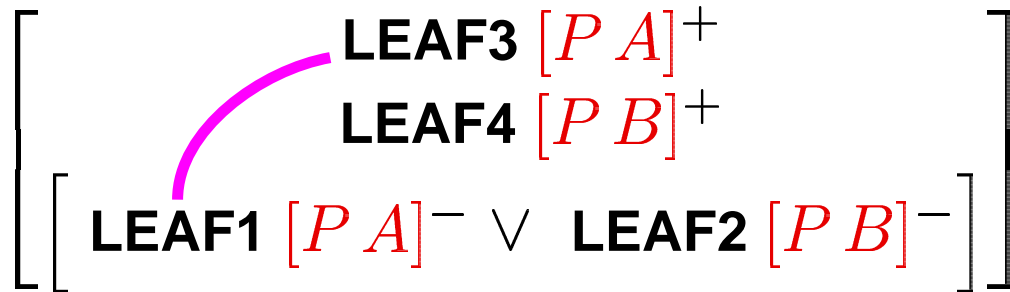
Span All VPs

Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

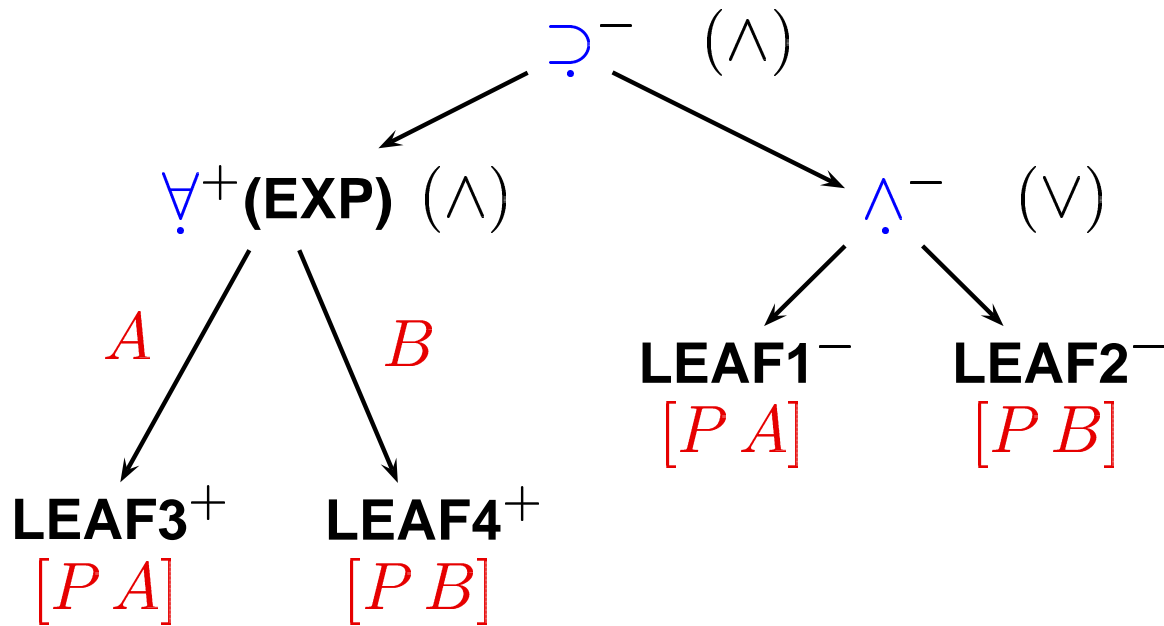


Complete Mating:
Span All VPs

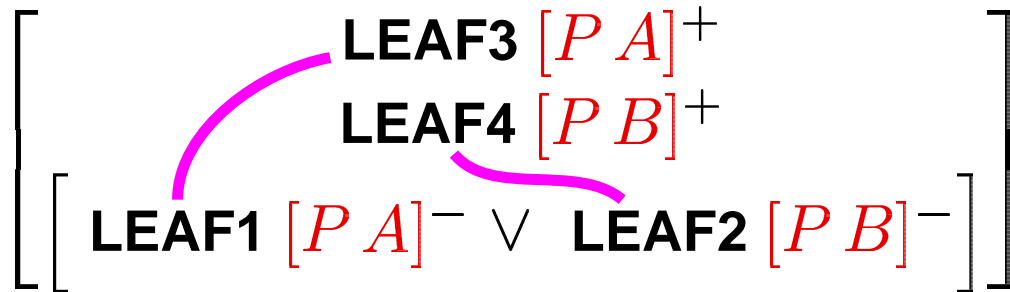


Expansion Trees: Simple Example 2

Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

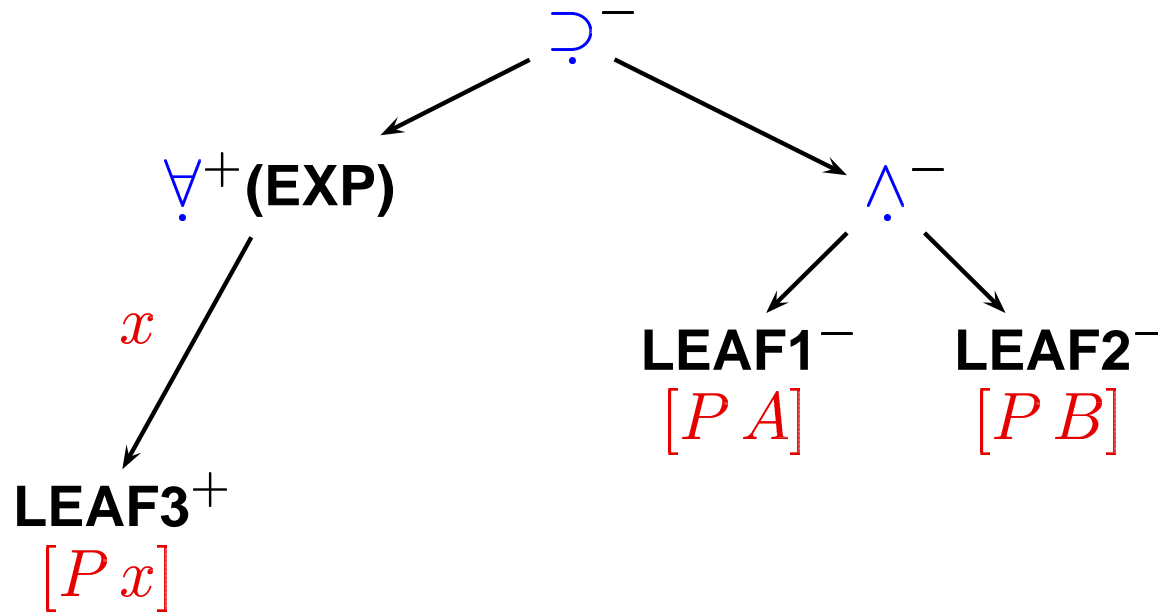


Complete Mating:
Span All VPs



Simple Search Example

Search Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

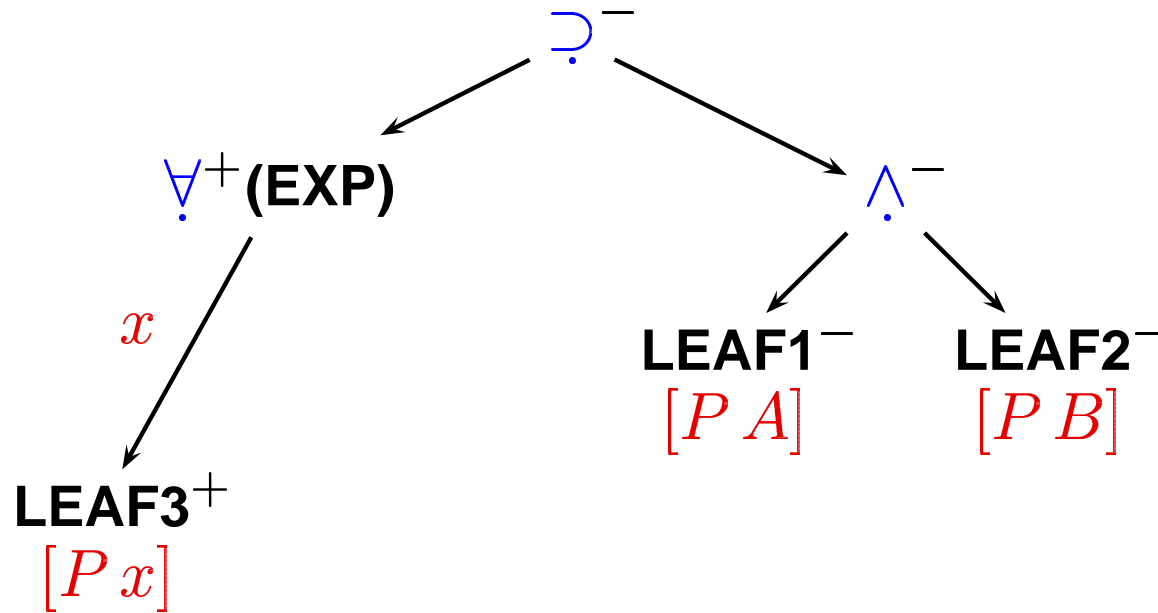


Expansion Var x
To Be Instantiated

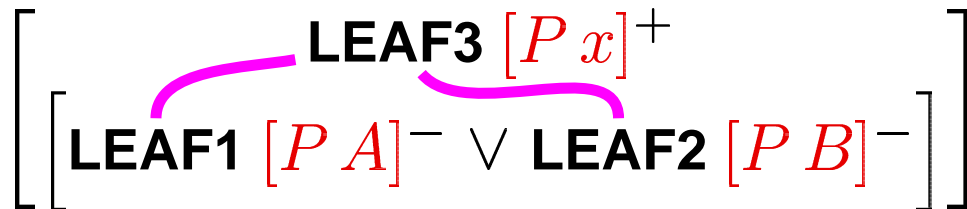
$$\left[\left[\text{LEAF1 } [P A]^{-} \vee \text{LEAF2 } [P B]^{-} \right] \right]$$

Simple Search Example

Search Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \cdot [P A_t] \wedge [P B_t]]$:

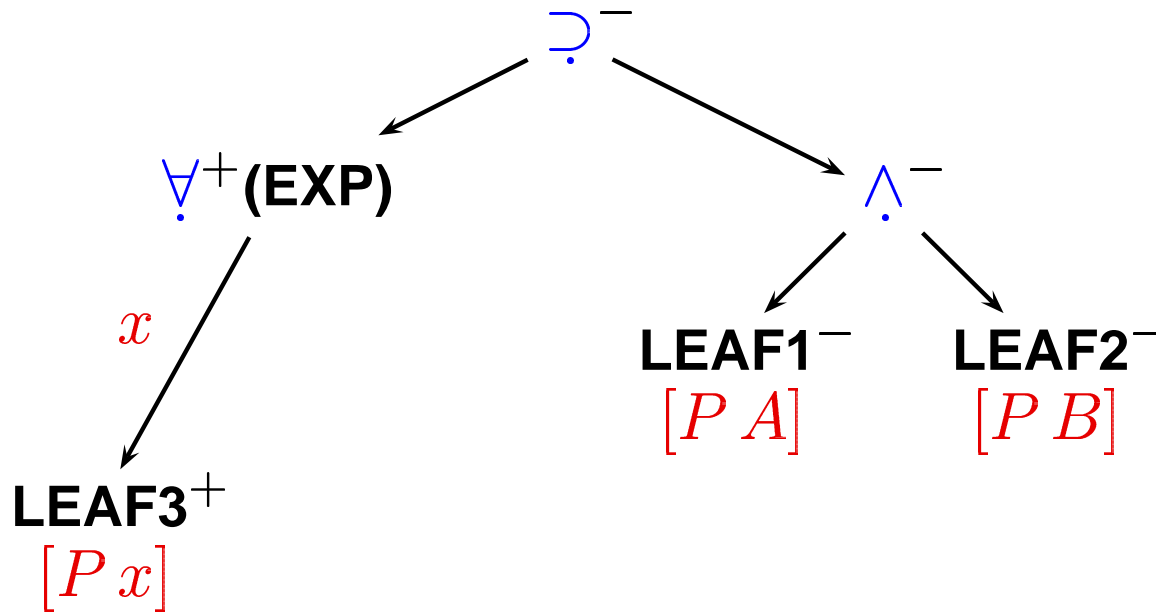


Complete Mating
But ...

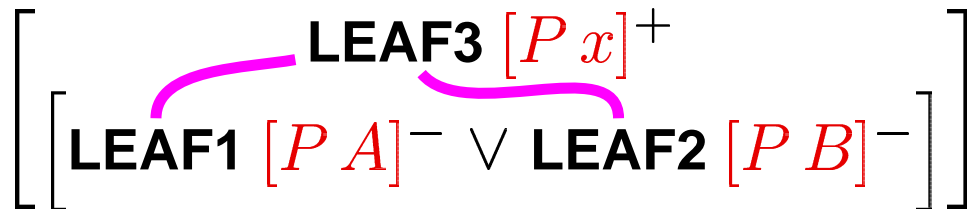


Simple Search Example

Search Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

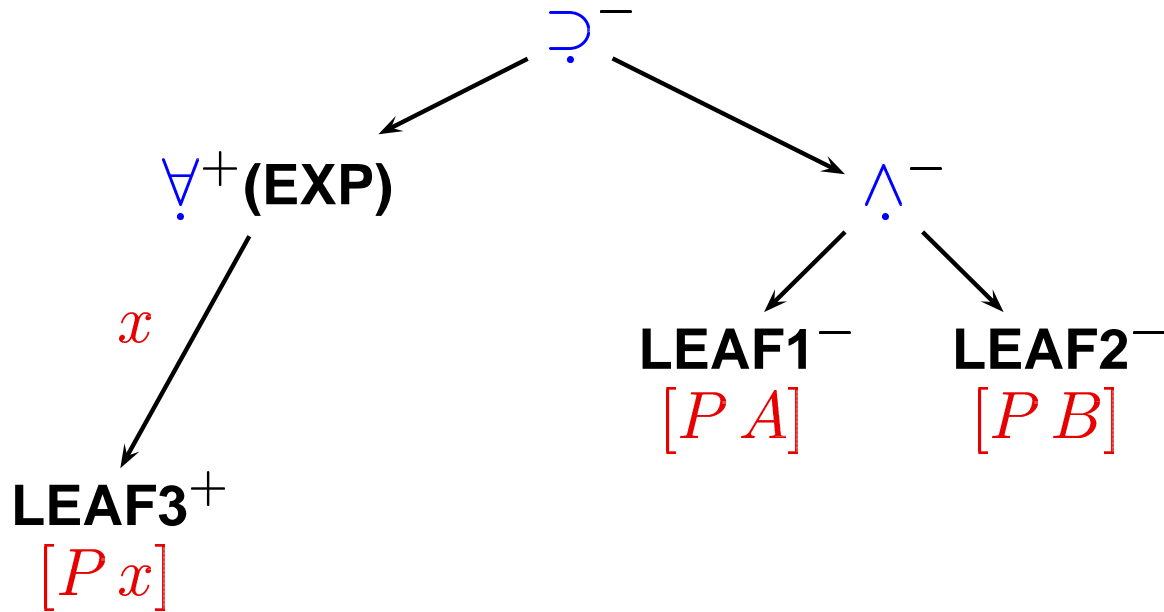


Can't Unify
 $x = ? A, x = ? B$

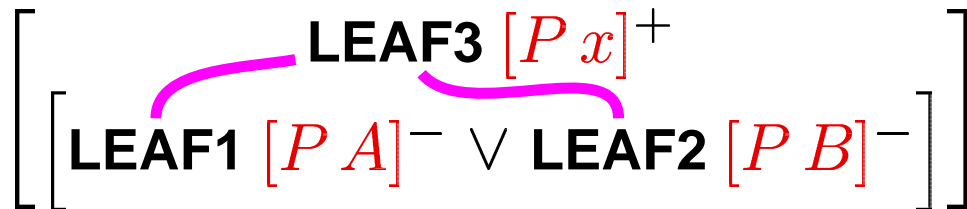


Simple Search Example

Search Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

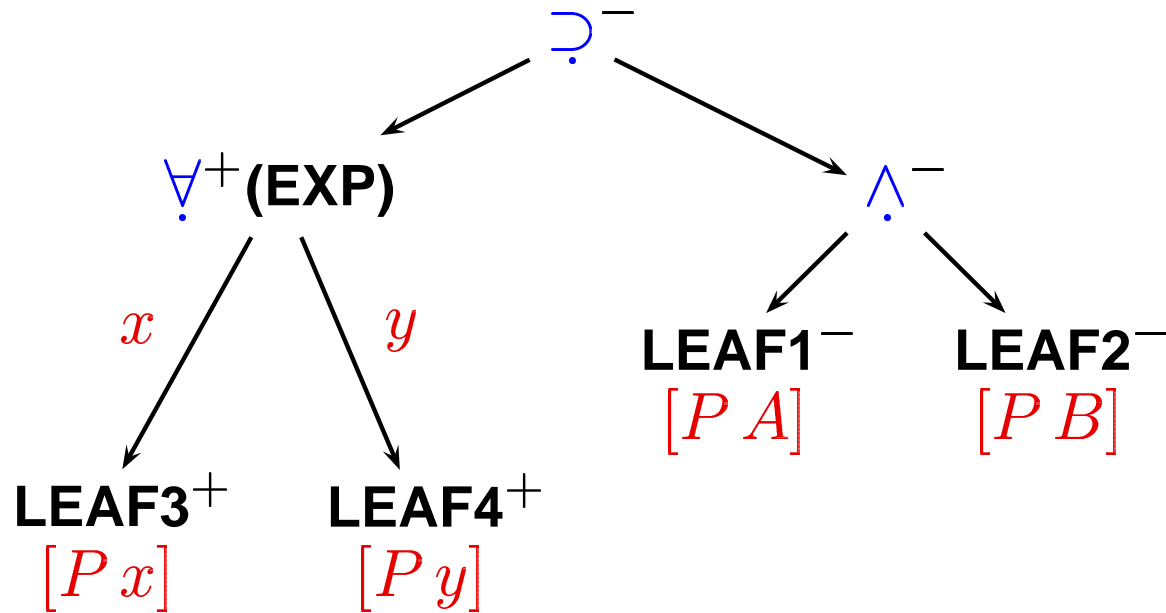


Duplicate x



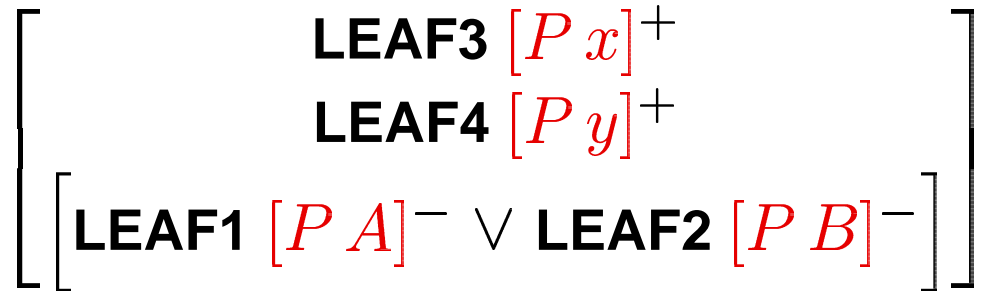
Simple Search Example

Search Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:



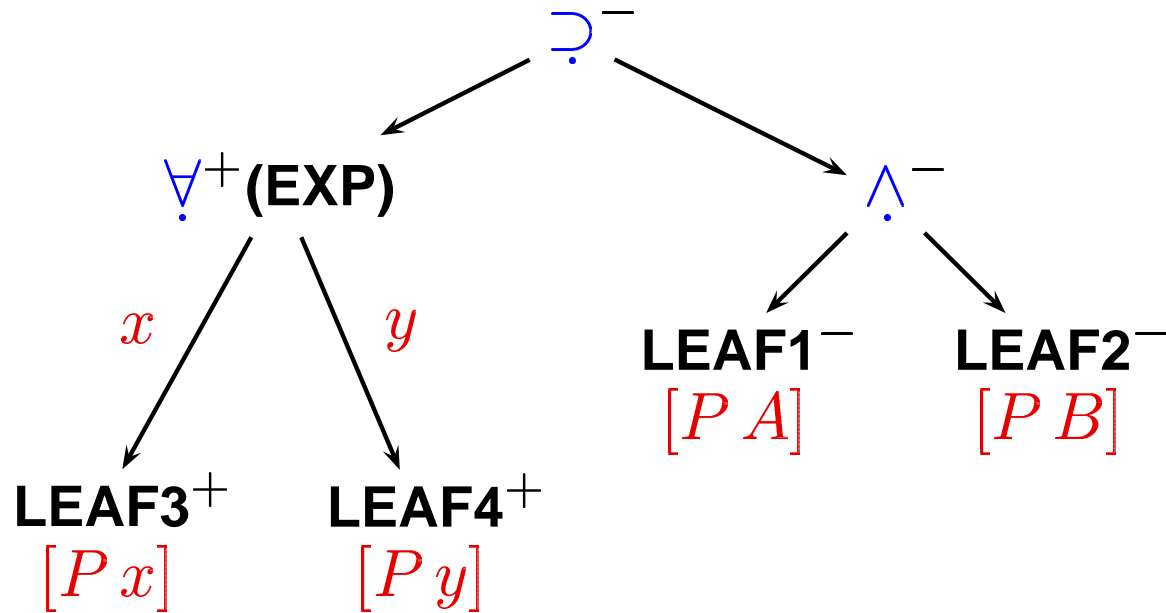
New Var y

New Tree and JForm

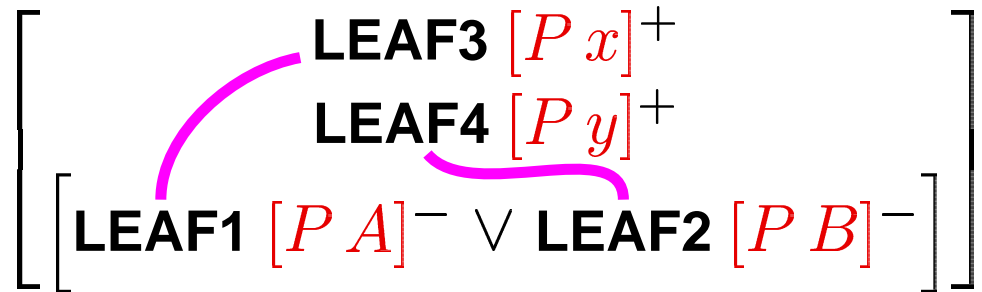


Simple Search Example

Search Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:

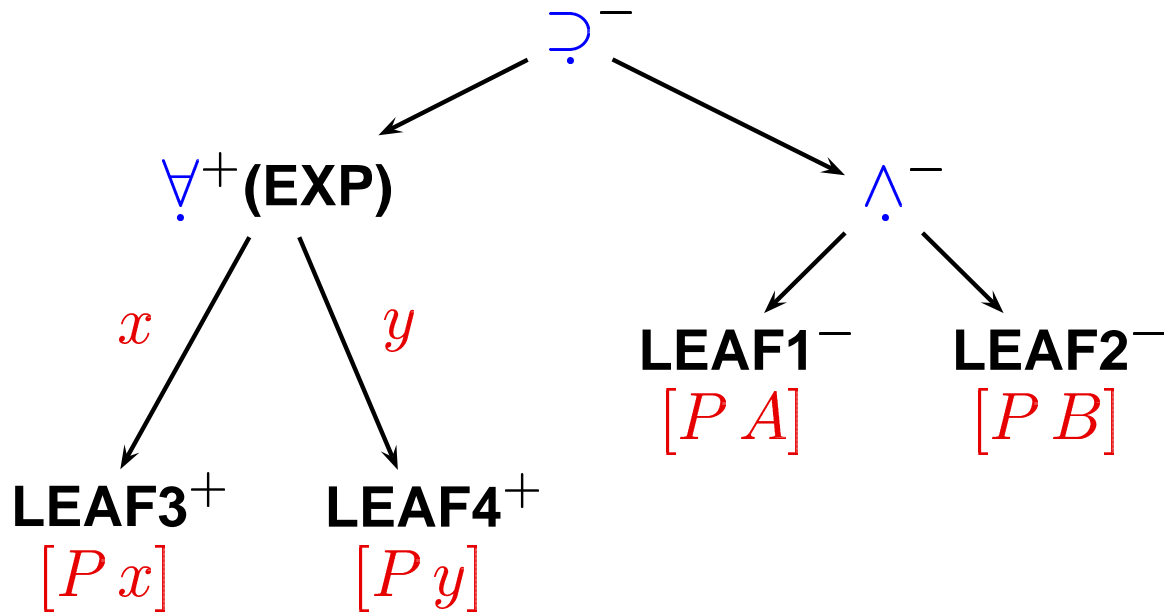


Complete Mating
And ...



Simple Search Example

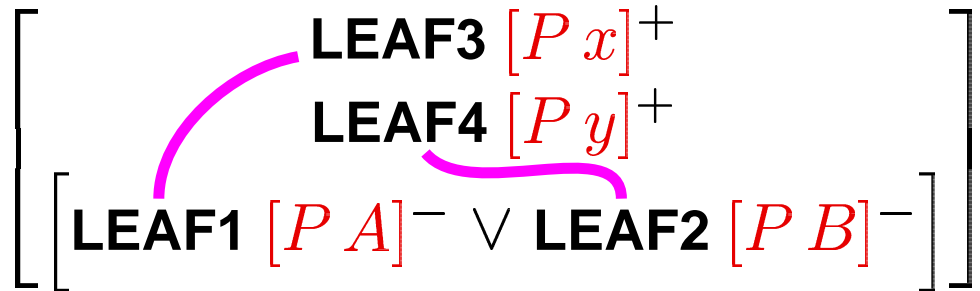
Search Expansion Tree for $[[\forall x_t [P_{ot} x]] \supset \neg [P A_t] \wedge [P B_t]]$:



$x \mapsto A, y \mapsto B$

Unify $x =? A, y =? B$

Success!



Search

- Find a Complete Mating

Search

- Find a Complete Mating
- Solve for Instantiations of Expansion Variables (Unification)

Search

- Find a Complete Mating
- Solve for Instantiations of Expansion Variables (Unification)
- Duplicate Expansions

Search

- Find a Complete Mating
- Solve for Instantiations of Expansion Variables (Unification)
- Duplicate Expansions
- Primsups - Introducing a Logical Constant

Higher-Order Example

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$

Higher-Order Example

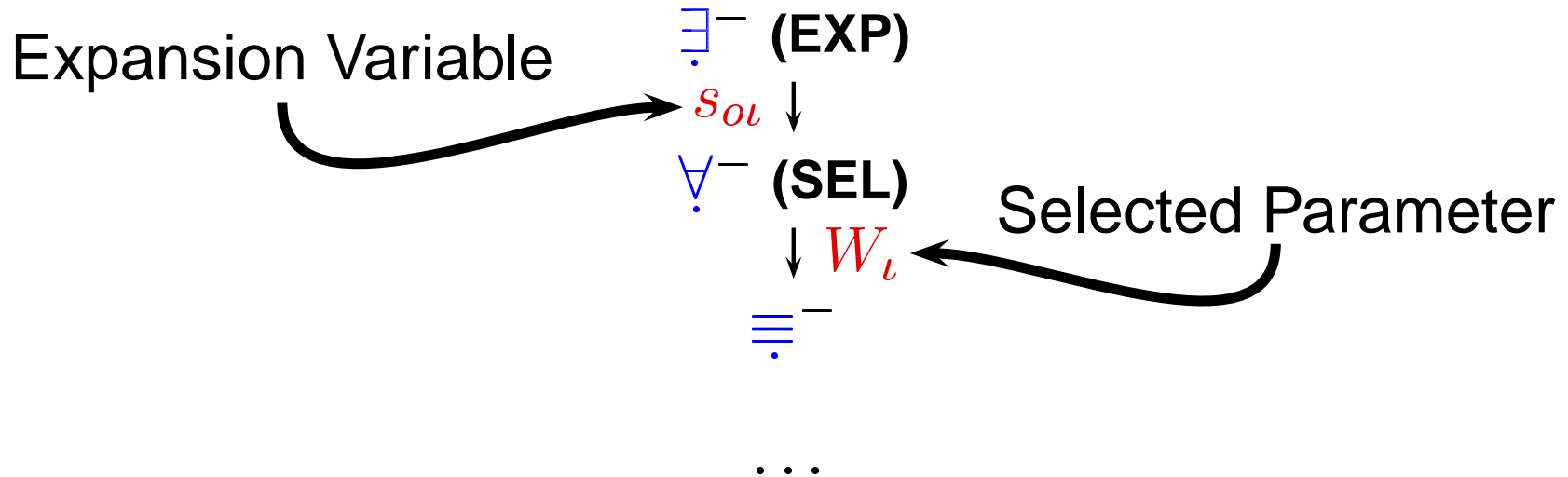
$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$

$s \mapsto A \cup B$ is the obvious solution.

I.E.: $s \mapsto [\lambda x_v. A x \vee B x]$, if $v \in \mathcal{S}$

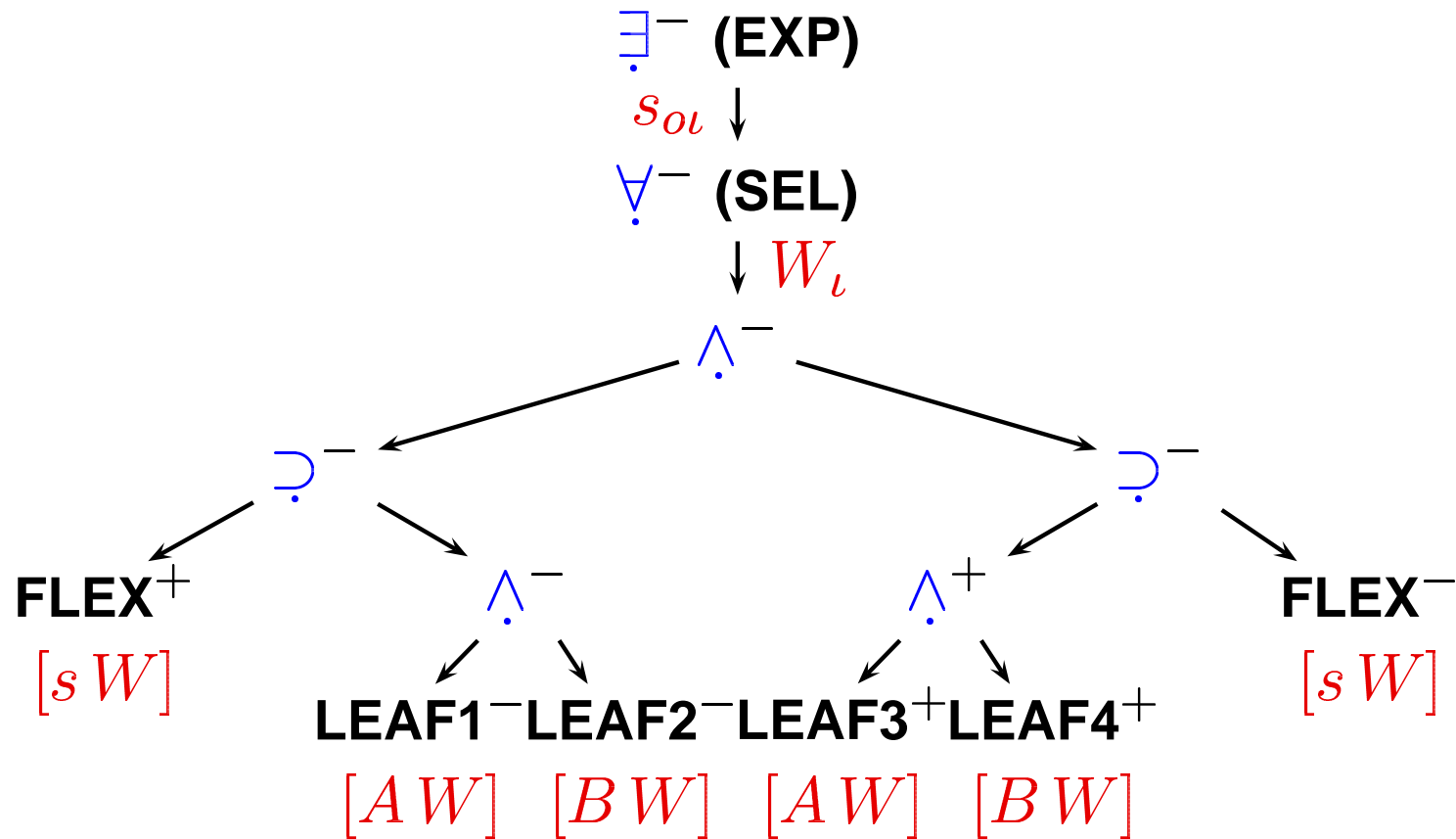
Higher-Order Example

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$



Higher-Order Example

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$



Higher-Order Example

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$

Primsusb

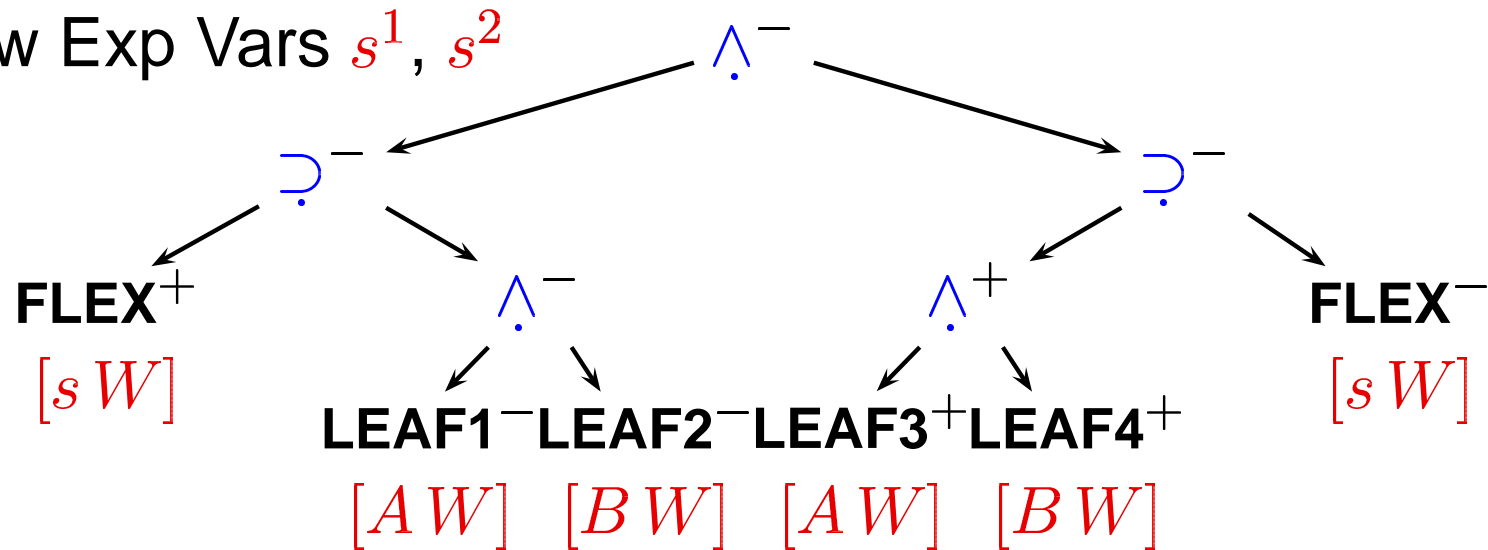
\exists^- (EXP)

$$s \mapsto [\lambda x_l \cdot s_{ol}^1 x \vee s_{ol}^2 x]$$

$s_{ol} \downarrow$
 \forall^- (SEL)

$\downarrow W_l$

New Exp Vars s^1, s^2



Higher-Order Example

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$

Primsusb

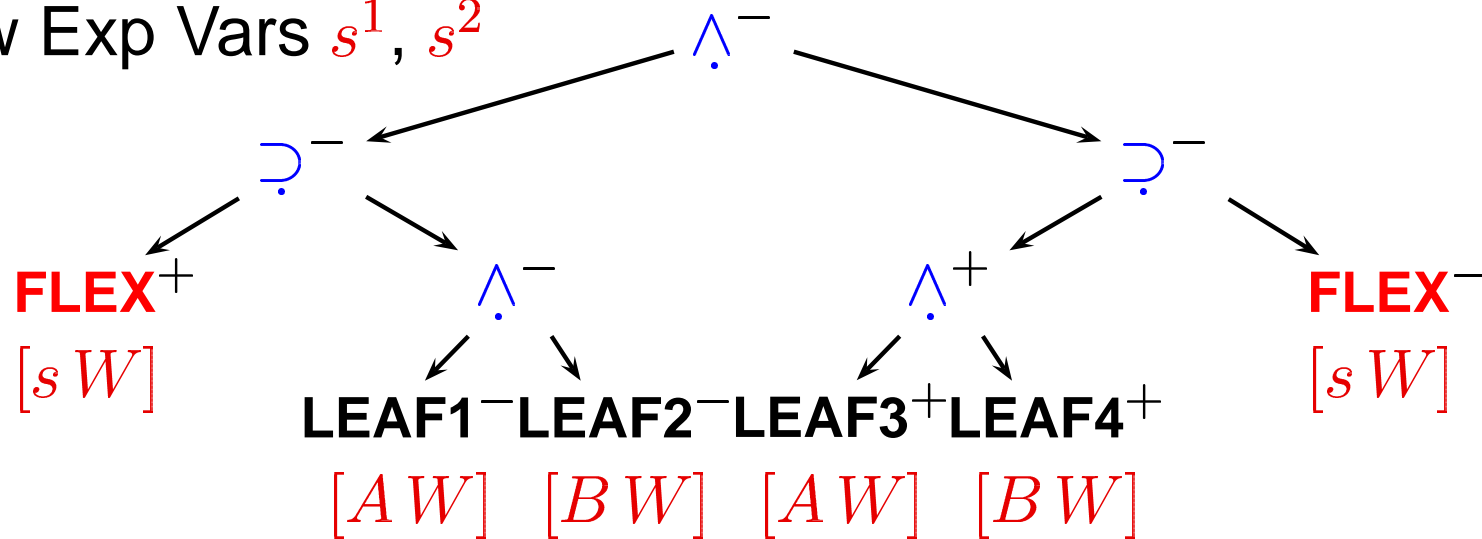
\exists^- (EXP)

$$s \mapsto [\lambda x_l \cdot s_{ol}^1 x \vee s_{ol}^2 x]$$

$s_{ol} \downarrow$
 \forall^- (SEL)

$\downarrow W_l$

New Exp Vars s^1, s^2



Higher-Order Example

$$\exists s_{ol} \forall Z_l [s Z \equiv \cdot A Z \vee B Z]$$

\exists^- (EXP)

$$[\lambda x_l \cdot s^1 x \vee s^2 x] \downarrow$$

\forall^- (SEL)

$$\downarrow W_l$$

\wedge^-

\supset^-

\supset^-

\vee^+

\wedge^-

\wedge^+

\vee^-

FLEX1⁺ FLEX2⁺ LEAF1⁻ LEAF2⁻ LEAF3⁺ LEAF4⁺ FLEX3⁻ FLEX4⁻

$[s^1 W] [s^2 W] [A W] [B W] [A W] [B W] [s^1 W] [s^2 W]$

Higher-Order Example

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$

\exists^- (EXP)

$$[\lambda x_v \cdot s^1 x \vee s^2 x] \downarrow$$

\forall^- (SEL)

$$\downarrow W_v$$

\wedge^-

\supset^-

\supset^-

\vee^+

\wedge^-

\wedge^+

\vee^-

FLEX1⁺ FLEX2⁺ LEAF1⁻ LEAF2⁻ LEAF3⁺ LEAF4⁺ FLEX3⁻ FLEX4⁻

$[s^1 W] [s^2 W] [A W] [B W] [A W] [B W] [s^1 W] [s^2 W]$



Higher-Order Example

$$\exists s_{ol} \forall Z_l [s Z \equiv \cdot A Z \vee B Z]$$

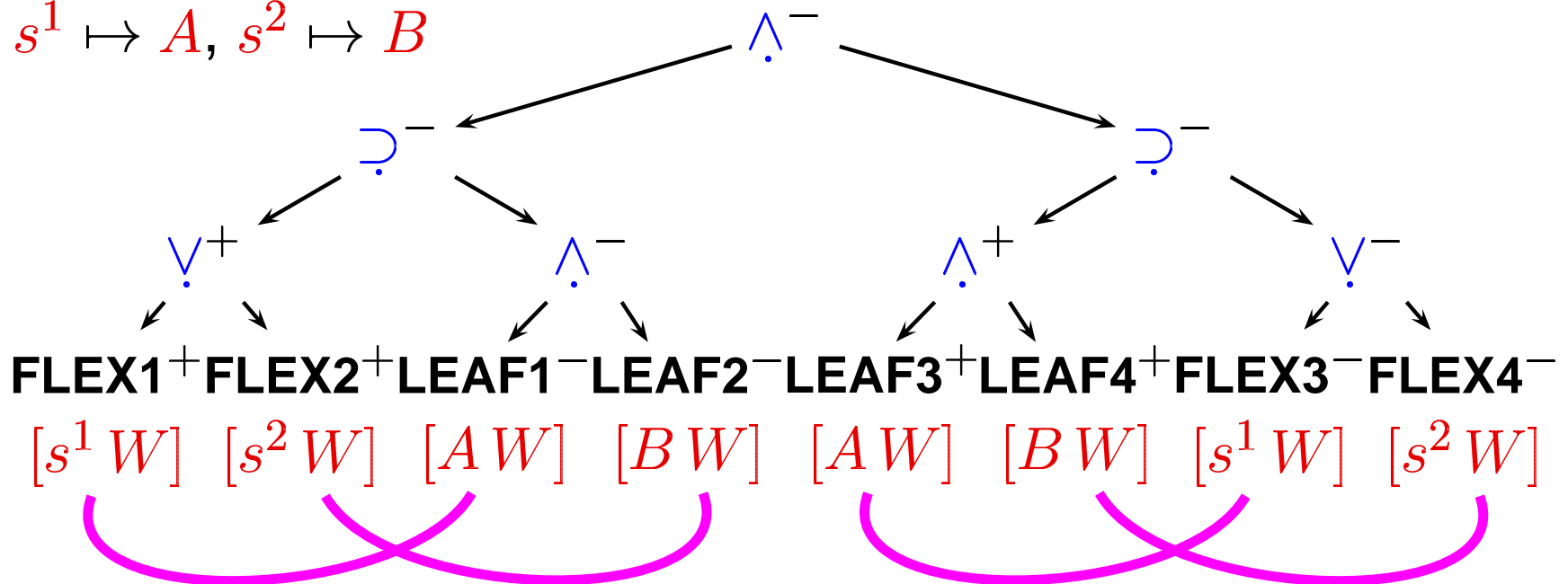
\exists^- (EXP)

$$[\lambda x_l \cdot s^1 x \vee s^2 x] \downarrow$$

\forall^- (SEL)

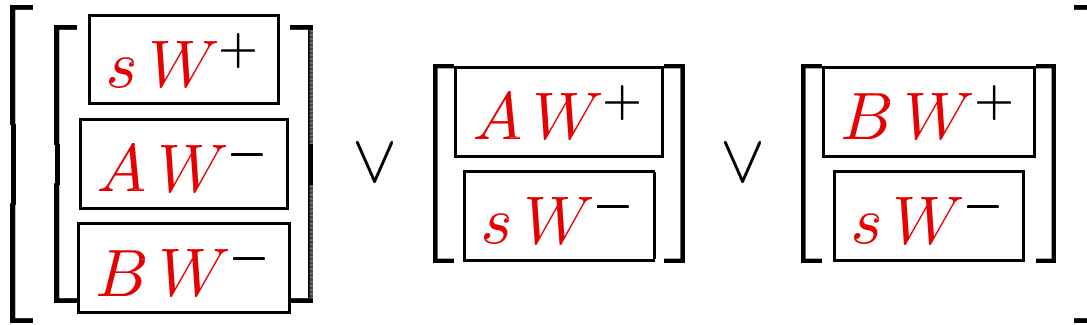
$\downarrow W_l$

$s^1 \mapsto A, s^2 \mapsto B$



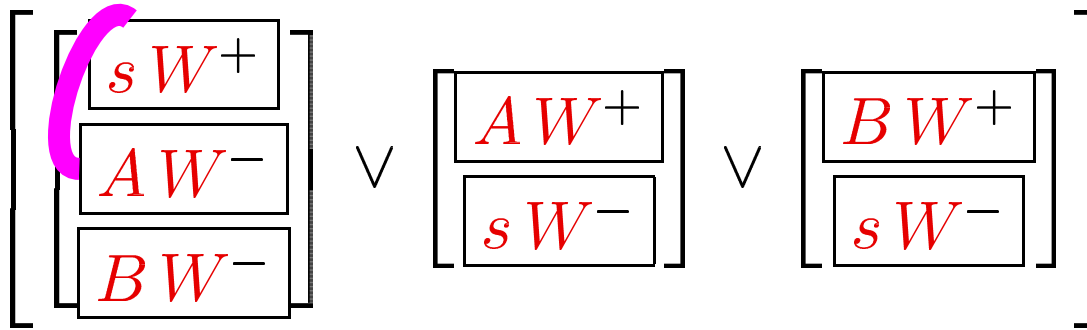
Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$



Higher-Order Example: JForms

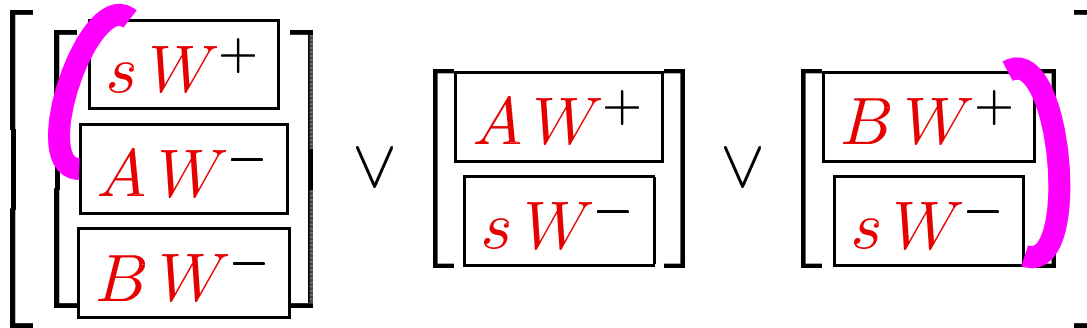
$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$



Requires: $s \mapsto A$

Higher-Order Example: JForms

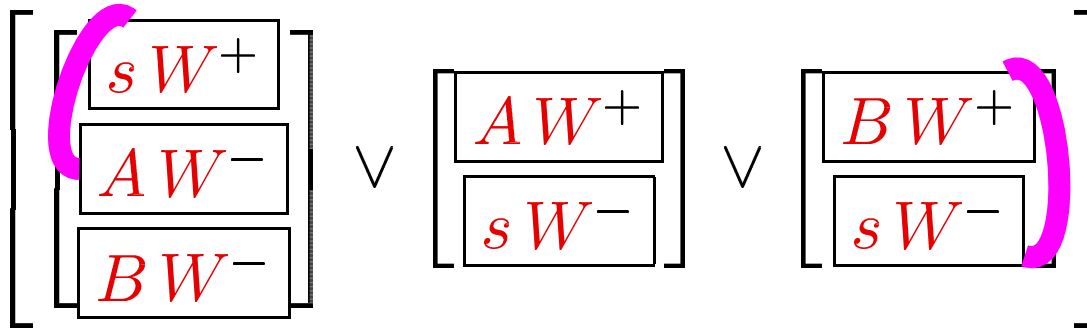
$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$



Requires: $s \mapsto A$ Requires: $s \mapsto B$

Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$

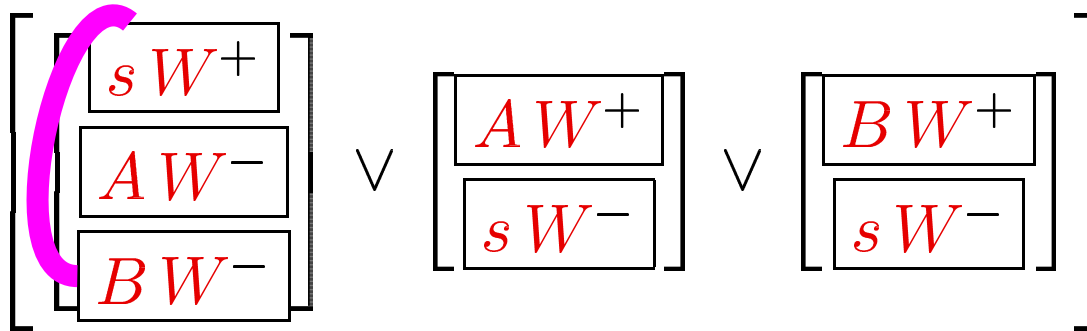


Requires: $s \mapsto A$ Requires: $s \mapsto B$

No Solution.

Higher-Order Example: JForms

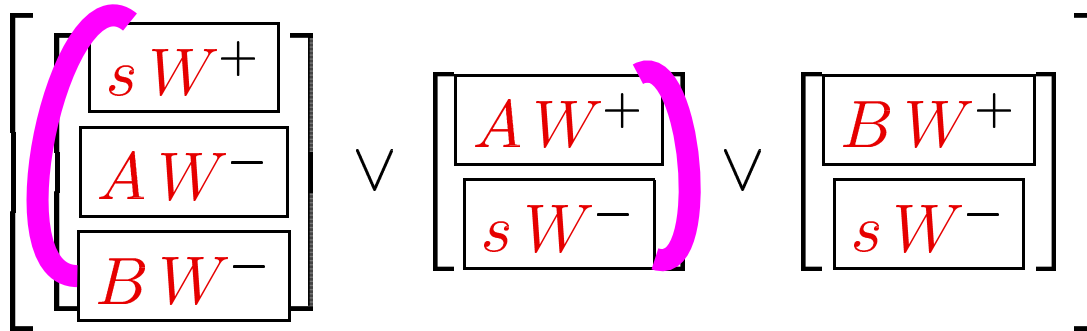
$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$



Requires: $s \mapsto B$

Higher-Order Example: JForms

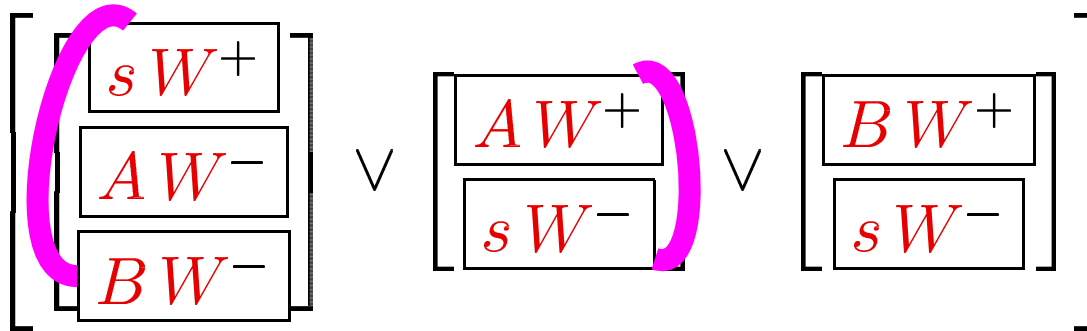
$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$



Requires: $s \mapsto B$ Requires: $s \mapsto A$

Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$



Requires: $s \mapsto B$ Requires: $s \mapsto A$

No Solution.

Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$

$$\left[\begin{array}{c} s W^+ \\ A W^- \\ B W^- \end{array} \vee \begin{array}{c} A W^+ \\ s W^- \end{array} \vee \begin{array}{c} B W^+ \\ s W^- \end{array} \right]$$

Need to Use Primsub $s \mapsto [\lambda x_v. s_{ol}^1 x \vee s_{ol}^2 x]$ to Introduce \vee .

Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$

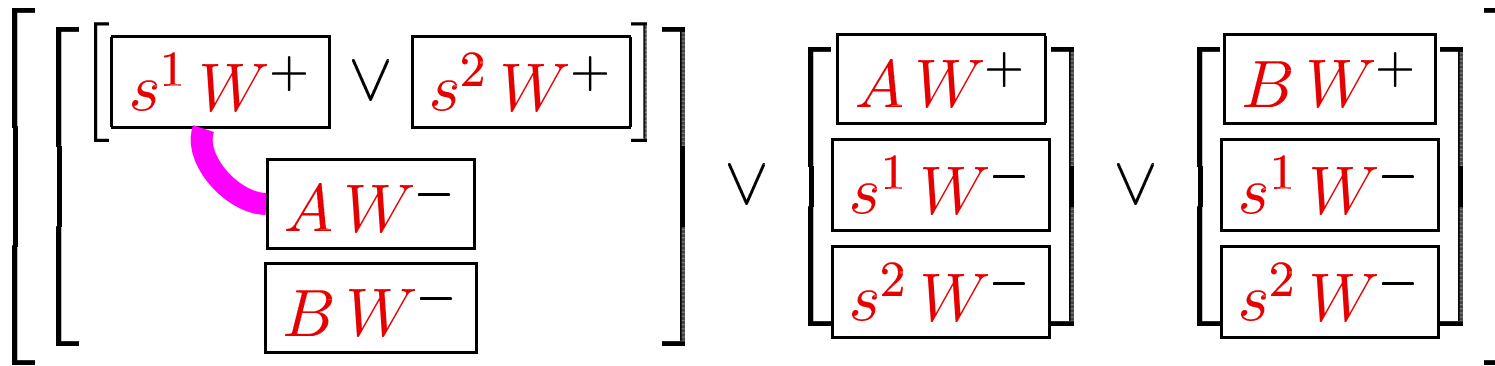
After Primsub for \vee :

$$\left[\left[\left[s^1 W^+ \vee s^2 W^+ \right] \right. \right. \\ \left. \left. \begin{array}{c} A W^- \\ B W^- \end{array} \right] \right] \vee \left[\begin{array}{c} A W^+ \\ s^1 W^- \\ s^2 W^- \end{array} \right] \vee \left[\begin{array}{c} B W^+ \\ s^1 W^- \\ s^2 W^- \end{array} \right]$$

Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$

After Primsub for \vee :

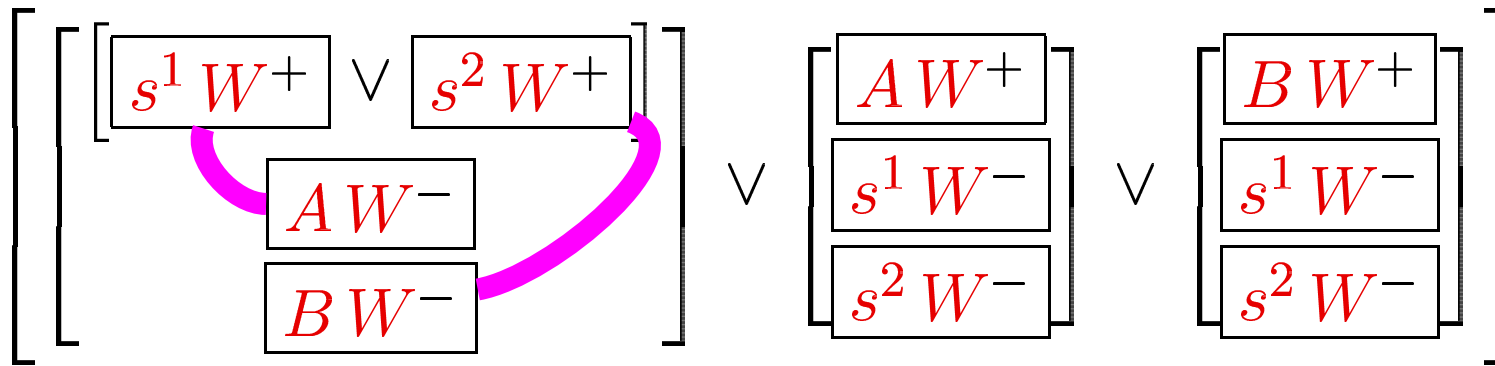


Requires: $s^1 \mapsto A$

Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$

After Primsub for \vee :

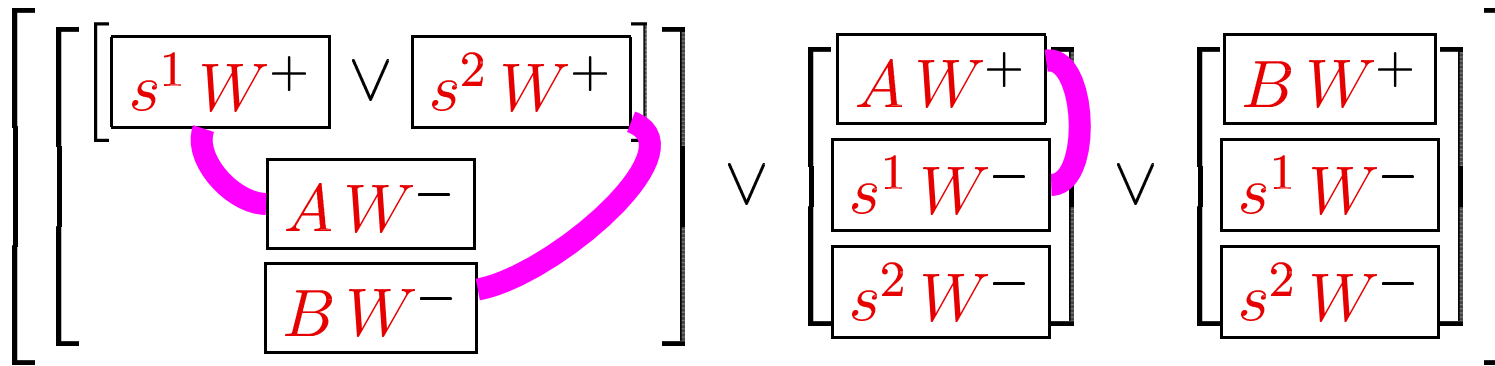


Requires: $s^1 \mapsto A$ Requires: $s^2 \mapsto B$

Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v [s Z \equiv \cdot A Z \vee B Z]$$

After Primsub for \vee :

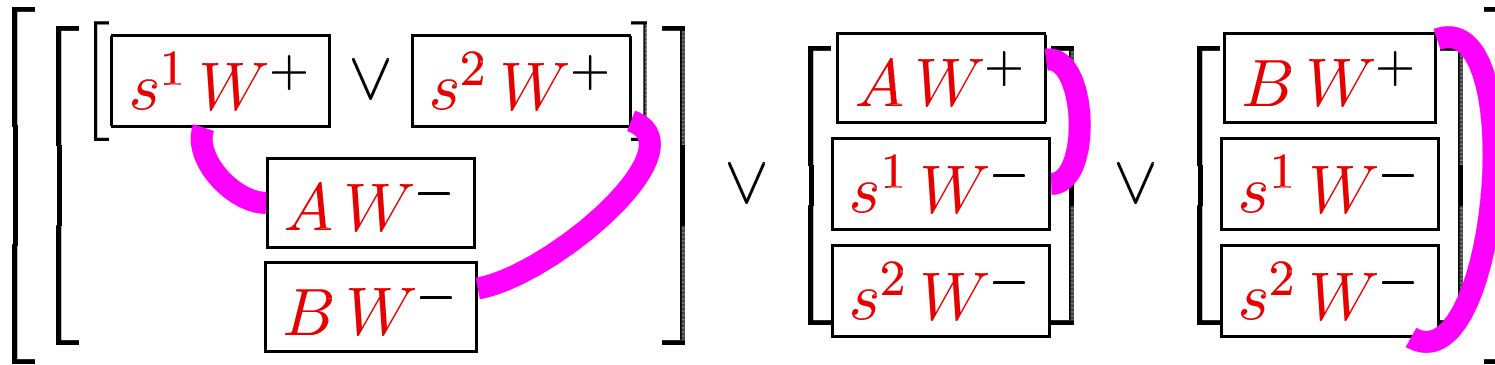


Requires: $s^1 \mapsto A$ Requires: $s^2 \mapsto B$

Higher-Order Example: JForms

$$\exists s_{ol} \forall Z_v. [s Z \equiv \cdot A Z \vee B Z]$$

After Primsub for \forall :



Requires: $s^1 \mapsto A$ Requires: $s^2 \mapsto B$

Complete Mating

Primitive Substitutions

Any set variable $s_{\alpha^n \dots \alpha^1}$ at the head of a flexible node may require a Primsub.

Primitive Substitutions

Any set variable $s_{\alpha^n \dots \alpha^1}$ at the head of a flexible node may require a Primsub.

- Primsubs: For each $c \in \mathcal{S}$

$$[\lambda x^1 \dots \lambda x^n [c [s^1 x^1 \dots x^n] \dots [s^m x^1 \dots x^n]]]$$

Primitive Substitutions

Any set variable $s_{o\alpha^n \dots \alpha^1}$ at the head of a flexible node may require a Primsub.

- Primsubs: For each $c \in \mathcal{S}$

$$[\lambda x^1 \dots \lambda x^n [c [s^1 x^1 \dots x^n] \dots [s^m x^1 \dots x^n]]]$$

- Projection Primsubs:

$$[\lambda x^1 \dots \lambda x^n [x^i [s^1 x^1 \dots x^n] \dots [s^m x^1 \dots x^n]]]$$

for each $1 \leq i \leq n$ where x^i is of some set type $(o\beta^m \dots \beta^1)$.

Primitive Substitutions

Any set variable $s_{o\alpha^n \dots \alpha^1}$ at the head of a flexible node may require a Primsub.

- Primsubs: For each $c \in \mathcal{S}$

$$[\lambda x^1 \dots \lambda x^n [c [s^1 x^1 \dots x^n] \dots [s^m x^1 \dots x^n]]]$$

- There may be infinitely many constants Π^α in \mathcal{S} .

Primitive Substitutions

Any set variable $s_{\alpha^n \dots \alpha^1}$ at the head of a flexible node may require a Primsub.

- Primsubs: For each $c \in \mathcal{S}$

$$[\lambda x^1 \dots \lambda x^n [c [s^1 x^1 \dots x^n] \dots [s^m x^1 \dots x^n]]]$$

- There may be infinitely many constants Π^α in \mathcal{S} .
- Can we restrict \mathcal{S} ?

Primitive Substitutions

Any set variable $s_{o\alpha^n \dots \alpha^1}$ at the head of a flexible node may require a Primsub.

- Primsubs: For each $c \in \mathcal{S}$

$$[\lambda x^1 \dots \lambda x^n [c [s^1 x^1 \dots x^n] \dots [s^m x^1 \dots x^n]]]$$

- There may be infinitely many constants Π^α in \mathcal{S} .
- Can we restrict \mathcal{S} ?
- Can we restrict the forms of instantiations \mathcal{S} ?

Primitive Substitutions

Any set variable $s_{o\alpha^n \dots \alpha^1}$ at the head of a flexible node may require a Primsub.

- Primsubs: For each $c \in \mathcal{S}$

$$[\lambda x^1 \dots \lambda x^n [c [s^1 x^1 \dots x^n] \dots [s^m x^1 \dots x^n]]]$$

- There may be infinitely many constants Π^α in \mathcal{S} .
- Can we restrict \mathcal{S} ?
- Can we restrict the forms of instantiations \mathcal{S} ?
- Difficult to Get *any* Restrictions without Extensionality

Boolean Extensionality

Let EXT^o be

$$\forall p_o \forall q_o \cdot [p \equiv q] \supset [p =^o q]$$

If two truth values are equivalent, then they are equal.

Functional Extensionality

Let $\mathbf{EXT}^{\alpha\beta}$ be

$$\forall f_{\alpha\beta} \forall g_{\alpha\beta} \cdot [\forall y_{\beta} [f y] =^{\alpha} [g y]] \supset [f =^{\alpha\beta} g]$$

If two functions agree on all arguments, then they are equal.

Functional Extensionality

Let $\text{EXT}^{\alpha\beta}$ be

$$\forall f_{\alpha\beta} \forall g_{\alpha\beta} \cdot [\forall y_{\beta} [f y] \doteq^{\alpha} [g y]] \supset [f \doteq^{\alpha\beta} g]$$

If two functions agree on all arguments, then they are equal.

Combining Boolean and Functional Extensionality:

$$\forall p_{o\alpha} \forall q_{o\alpha} \cdot [\forall x_{\alpha} [p x] \equiv [q x]] \supset [p \doteq^{o\alpha} q]$$

Two sets are equal if they contain the same elements.

Without Extensionality

- Without Extensionality $\neg\neg A_0$ May Differ from A .

Without Extensionality

- Without Extensionality $\neg\neg A_o$ May Differ from A .
- The Sets $[\lambda x_\alpha \neg\neg A]$ and $[\lambda x_\alpha A]$ May Differ.

Without Extensionality

- Without Extensionality $\neg\neg\mathbf{A}_o$ May Differ from \mathbf{A} .
- The Sets $[\lambda x_\alpha \neg\neg\mathbf{A}]$ and $[\lambda x_\alpha \mathbf{A}]$ May Differ.
- We May Need to Consider a Set Instantiation $[\lambda x_\alpha \neg\neg\mathbf{A}]$.

Without Extensionality

- Without Extensionality $\neg\neg A_o$ May Differ from A .
- The Sets $[\lambda x_\alpha \neg\neg A]$ and $[\lambda x_\alpha A]$ May Differ.
- We May Need to Consider a Set Instantiation $[\lambda x_\alpha \neg\neg A]$.
- $[\forall p_o \cdot Q_{oop}]$ May Differ from $[[Q \top] \wedge [Q \perp]]$.

Without Extensionality

- Without Extensionality $\neg\neg\mathbf{A}_o$ May Differ from \mathbf{A} .
- The Sets $[\lambda x_\alpha \neg\neg\mathbf{A}]$ and $[\lambda x_\alpha \mathbf{A}]$ May Differ.
- We May Need to Consider a Set Instantiation $[\lambda x_\alpha \neg\neg\mathbf{A}]$.
- $[\forall p_o \cdot \mathbf{Q}_{oop}]$ May Differ from $[[\mathbf{Q} \top] \wedge [\mathbf{Q} \perp]]$.
- We May Need to Consider Logical Constant Π^o in a Proof.

Extensional Theorems

Let M be the sentence

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

Extensional Theorems

Let M be the sentence

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

- T_{PS} traditionally proves theorems of elementary type theory (no extensionality).

Extensional Theorems

Let M be the sentence

$$[A_o \supset \neg B_o \supset \neg [P_{oo} A_o] \supset [P_{oo} B_o]]$$

- T_{PS} traditionally proves theorems of elementary type theory (no extensionality).
- Previous T_{PS} search procedures cannot prove M .

Extensional Theorems

Let M be the sentence

$$[A_o \supset \neg B_o \supset \neg [P_{oo} A_o] \supset [P_{oo} B_o]]$$

- TPS traditionally proves theorems of elementary type theory (no extensionality).
- Previous TPS search procedures cannot prove M .
- Can simply include Boolean extensionality as a hypothesis $EXT^o \supset M$.

Extensional Theorems

Let M be the sentence

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

- T_{PS} traditionally proves theorems of elementary type theory (no extensionality).
- Previous T_{PS} search procedures cannot prove M .
- Can simply include Boolean extensionality as a hypothesis $EXT^o \supset M$.
- Drawback: Using

$$\forall p_o \forall q_o \cdot [p \equiv q] \supset [p \overset{o}{=} q]$$

in the hypothesis requires instantiating p_o and q_o .

Extensional Theorems

Let M be the sentence

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

- T_{PS} traditionally proves theorems of elementary type theory (no extensionality).
- Previous T_{PS} search procedures cannot prove M .
- Can simply include Boolean extensionality as a hypothesis $EXT^o \supset M$.
- Drawback: Using

$$\forall p_o \forall q_o \cdot [p \equiv q] \supset [p \doteq^o q]$$

in the hypothesis requires instantiating p_o and q_o .

- Want: A better way to build in extensionality.

Sequents

Sequents = Finite Multiset of Sentences

$$M^1, \dots, M^n$$

Sequents

Sequents = Finite Multiset of Sentences

$$M^1, \dots, M^n$$

Intuitively,

$$M^1 \vee \dots \vee M^n$$

Sequents

Sequents = Finite Multiset of Sentences

$$M^1, \dots, M^n$$

Intuitively,

$$M^1 \vee \dots \vee M^n$$

Next: Rules for Deriving Sequents
in Extensional Type Theory

Basic Sequent Rules

Γ, M

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{}{\Gamma, \mathbf{M}}$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \perp}$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \perp} \mathcal{G}(\top)$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \perp} \mathcal{G}(\top)$$

$$\frac{}{\Gamma, \neg\neg\mathbf{M}}$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \perp} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \top} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

$$\frac{}{\Gamma, [\mathbf{M} \vee \mathbf{N}]}$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \top} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

$$\frac{\Gamma, \mathbf{M}, \mathbf{N}}{\Gamma, [\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\vee)$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \top} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

$$\frac{\Gamma, \mathbf{M}, \mathbf{N}}{\Gamma, [\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\vee)$$

$$\frac{}{\Gamma, \neg[\mathbf{M} \vee \mathbf{N}]}$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \top} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

$$\frac{\Gamma, \mathbf{M}, \mathbf{N}}{\Gamma, [\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\vee)$$

$$\frac{\Gamma, \neg\mathbf{M} \quad \Gamma, \neg\mathbf{N}}{\Gamma, \neg[\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\neg\vee)$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \top} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

$$\frac{\Gamma, \mathbf{M}, \mathbf{N}}{\Gamma, [\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\vee)$$

$$\frac{\Gamma, \neg\mathbf{M} \quad \Gamma, \neg\mathbf{N}}{\Gamma, \neg[\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\neg\vee)$$

$$\Gamma, [\forall x_\alpha \mathbf{M}]$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \top} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

$$\frac{\Gamma, \mathbf{M}, \mathbf{N}}{\Gamma, [\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\vee)$$

$$\frac{\Gamma, \neg\mathbf{M} \quad \Gamma, \neg\mathbf{N}}{\Gamma, \neg[\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\neg\vee)$$

$$\frac{\Gamma, [[W/x]\mathbf{M}] \quad W \in \mathcal{P}_\alpha \setminus \mathbf{Params}(\Gamma, [\forall x_\alpha \mathbf{M}])}{\Gamma, [\forall x_\alpha \mathbf{M}]} \mathcal{G}(\forall^W)$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \top} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

$$\frac{\Gamma, \mathbf{M}, \mathbf{N}}{\Gamma, [\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\vee)$$

$$\frac{\Gamma, \neg\mathbf{M} \quad \Gamma, \neg\mathbf{N}}{\Gamma, \neg[\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\neg\vee)$$

$$\frac{\Gamma, [[W/x]\mathbf{M}] \quad W \in \mathcal{P}_\alpha \setminus \mathbf{Params}(\Gamma, [\forall x_\alpha \mathbf{M}])}{\Gamma, [\forall x_\alpha \mathbf{M}]} \mathcal{G}(\forall^W)$$

$$\Gamma, \neg[\forall x_\alpha \mathbf{M}]$$

Basic Sequent Rules

$$\frac{\Gamma, \mathbf{M}, \mathbf{M}}{\Gamma, \mathbf{M}} \mathcal{G}(\text{Contr})$$

$$\frac{\Gamma, \mathbf{M}^\downarrow}{\Gamma, \mathbf{M}} \mathcal{G}(\beta\eta)$$

$$\frac{}{\Gamma, \top} \mathcal{G}(\top)$$

$$\frac{\Gamma, \mathbf{M}}{\Gamma, \neg\neg\mathbf{M}} \mathcal{G}(\neg\neg)$$

$$\frac{\Gamma, \mathbf{M}, \mathbf{N}}{\Gamma, [\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\vee)$$

$$\frac{\Gamma, \neg\mathbf{M} \quad \Gamma, \neg\mathbf{N}}{\Gamma, \neg[\mathbf{M} \vee \mathbf{N}]} \mathcal{G}(\neg\vee)$$

$$\frac{\Gamma, [[\mathbf{W}/x]\mathbf{M}] \quad \mathbf{W} \in \mathcal{P}_\alpha \setminus \mathbf{Params}(\Gamma, [\forall x_\alpha \mathbf{M}])}{\Gamma, [\forall x_\alpha \mathbf{M}]} \mathcal{G}(\forall^{\mathbf{W}})$$

$$\frac{\Gamma, \neg[[\mathbf{C}/x]\mathbf{M}] \quad \mathbf{C} \in \text{cwff}_\alpha(\mathcal{S})}{\Gamma, \neg[\forall x_\alpha \mathbf{M}]} \mathcal{G}(\neg\forall, \mathcal{S})$$

Rules for Logical Constants

 $c \in \mathcal{S}$

 $\Gamma, [c \overline{A}^n]$ $c \in \mathcal{S}$

 $\Gamma, \neg [c \overline{A}^n]$

Rules for Logical Constants

$$\frac{\Gamma, [c \overline{\mathbf{A}^n}]^\# \quad c \in \mathcal{S}}{\Gamma, [c \mathbf{A}^n]} \mathcal{G}(\#)$$

$$\frac{\Gamma, \neg([c \overline{\mathbf{A}^n}]^\#) \quad c \in \mathcal{S}}{\Gamma, \neg[c \mathbf{A}^n]} \mathcal{G}(\neg\#)$$

Rules for Logical Constants

$$\frac{\Gamma, [c \overline{\mathbf{A}^n}]^\# \quad c \in \mathcal{S}}{\Gamma, [c \mathbf{A}^n]} \mathcal{G}(\#)$$

$$\frac{\Gamma, \neg([c \overline{\mathbf{A}^n}]^\#) \quad c \in \mathcal{S}}{\Gamma, \neg[c \mathbf{A}^n]} \mathcal{G}(\neg\#)$$

$$\top^\# := \perp$$

Rules for Logical Constants

$$\frac{\Gamma, [c \overline{\mathbf{A}^n}]^\# \quad c \in \mathcal{S}}{\Gamma, [c \overline{\mathbf{A}^n}]} \mathcal{G}(\#)$$

$$\frac{\Gamma, \neg([c \overline{\mathbf{A}^n}]^\#) \quad c \in \mathcal{S}}{\Gamma, \neg[c \overline{\mathbf{A}^n}]} \mathcal{G}(\neg\#)$$

$$\top^\# := \perp$$

$$\perp^\# := \neg\perp$$

Rules for Logical Constants

$$\frac{\Gamma, [c \overline{\mathbf{A}^n}]^\# \quad c \in \mathcal{S}}{\Gamma, [c \overline{\mathbf{A}^n}]} \mathcal{G}(\#)$$

$$\frac{\Gamma, \neg([c \overline{\mathbf{A}^n}]^\#) \quad c \in \mathcal{S}}{\Gamma, \neg[c \overline{\mathbf{A}^n}]} \mathcal{G}(\neg\#)$$

$$\top^\# := \top$$

$$\perp^\# := \neg\top$$

$$[\neg \mathbf{A}_o]^\# := \neg \mathbf{A}_o$$

Rules for Logical Constants

$$\frac{\Gamma, [c \overline{\mathbf{A}^n}]^\# \quad c \in \mathcal{S}}{\Gamma, [c \overline{\mathbf{A}^n}]} \mathcal{G}(\#)$$

$$\frac{\Gamma, \neg([c \overline{\mathbf{A}^n}]^\#) \quad c \in \mathcal{S}}{\Gamma, \neg[c \overline{\mathbf{A}^n}]} \mathcal{G}(\neg\#)$$

$$\top^\# := \top$$

$$\perp^\# := \neg\top$$

$$[\neg \mathbf{A}_o]^\# := \neg \mathbf{A}_o$$

$$[\mathbf{A}_o \vee \mathbf{B}_o]^\# := [\mathbf{A}_o \vee \mathbf{B}_o]$$

Rules for Logical Constants

$$\frac{\Gamma, [c \overline{\mathbf{A}^n}]^\# \quad c \in \mathcal{S}}{\Gamma, [c \overline{\mathbf{A}^n}]} \mathcal{G}(\#)$$

$$\frac{\Gamma, \neg([c \overline{\mathbf{A}^n}]^\#) \quad c \in \mathcal{S}}{\Gamma, \neg[c \overline{\mathbf{A}^n}]} \mathcal{G}(\neg\#)$$

$$\top^\# := \top$$

$$\perp^\# := \neg\top$$

$$[\neg \mathbf{A}_o]^\# := \neg \mathbf{A}_o$$

$$[\mathbf{A}_o \vee \mathbf{B}_o]^\# := [\mathbf{A}_o \vee \mathbf{B}_o]$$

$$[\mathbf{A}_o \supset \mathbf{B}_o]^\# := [\neg \mathbf{A}_o \vee \mathbf{B}_o]$$

Rules for Logical Constants

$$\frac{\Gamma, [c \overline{\mathbf{A}^n}]^\# \quad c \in \mathcal{S}}{\Gamma, [c \overline{\mathbf{A}^n}]} \mathcal{G}(\#)$$

$$\frac{\Gamma, \neg([c \overline{\mathbf{A}^n}]^\#) \quad c \in \mathcal{S}}{\Gamma, \neg[c \overline{\mathbf{A}^n}]} \mathcal{G}(\neg\#)$$

$$\top^\# := \top$$

$$\perp^\# := \neg\top$$

$$[\neg \mathbf{A}_o]^\# := \neg \mathbf{A}_o$$

$$[\mathbf{A}_o \vee \mathbf{B}_o]^\# := [\mathbf{A}_o \vee \mathbf{B}_o]$$

$$[\mathbf{A}_o \supset \mathbf{B}_o]^\# := [\neg \mathbf{A}_o \vee \mathbf{B}_o]$$

$$[\Pi^\alpha \cdot \mathbf{F}_{o\alpha}]^\# := [\forall x_\alpha \cdot \mathbf{F}_{o\alpha} x_\alpha]$$

Rules for Logical Constants

$$\frac{\Gamma, [c \overline{\mathbf{A}^n}]^\# \quad c \in \mathcal{S}}{\Gamma, [c \overline{\mathbf{A}^n}]} \mathcal{G}(\#) \qquad \frac{\Gamma, \neg([c \overline{\mathbf{A}^n}]^\#) \quad c \in \mathcal{S}}{\Gamma, \neg[c \overline{\mathbf{A}^n}]} \mathcal{G}(\neg\#)$$

$$\top^\# := \top \qquad \perp^\# := \neg\top \qquad [\neg \mathbf{A}_o]^\# := \neg \mathbf{A}_o$$

$$[\mathbf{A}_o \vee \mathbf{B}_o]^\# := [\mathbf{A}_o \vee \mathbf{B}_o] \qquad [\mathbf{A}_o \supset \mathbf{B}_o]^\# := [\neg \mathbf{A}_o \vee \mathbf{B}_o]$$

$$[\Pi^\alpha \bullet \mathbf{F}_{o\alpha}]^\# := [\forall x_\alpha \bullet \mathbf{F}_{o\alpha} x_\alpha]$$

$$[\mathbf{A}_\alpha =^\alpha \mathbf{B}_\alpha]^\# := [\mathbf{A}_\alpha \doteq^\alpha \mathbf{B}_\alpha]$$

Initial Rule (up to Equality)

$$\frac{\mathbf{A} \in \text{cwff}_o(\mathcal{S}) \text{ atom}}{\Gamma, \mathbf{A}, \neg \mathbf{A}} \mathcal{G}(\text{Init})$$

What about $[P \top], \neg[P \cdot \neg \perp]$?

Initial Rule (up to Equality)

$$\frac{A \in \text{cwff}_0(S) \text{ atom}}{\Gamma, A, \dots \vdash A} \mathcal{G}(\text{Init})$$

What about $[P \top], \neg[P \cdot \neg \perp]$?

Initial Rule (up to Equality)

$n \geq 0 :$

$$\frac{\Gamma, [\mathbf{A}^1 \doteq \mathbf{B}^1] \quad \dots \quad \Gamma, [\mathbf{A}^n \doteq \mathbf{B}^n] \quad P_{o\alpha^n \dots \alpha^1} \in \mathcal{P}}{\Gamma, [P \overline{\mathbf{A}^n}], \neg [P \overline{\mathbf{B}^n}]} \mathcal{G}(\text{Init}^{\overline{\cdot}})$$

Initial Rule (up to Equality)

$n \geq 0 :$

$$\frac{\Gamma, [\mathbf{A}^1 = \mathbf{B}^1] \quad \dots \quad \Gamma, [\mathbf{A}^n = \mathbf{B}^n] \quad P_{o\alpha^n \dots \alpha^1} \in \mathcal{P}}{\Gamma, [P \overline{\mathbf{A}^n}], \neg [P \overline{\mathbf{B}^n}]} \mathcal{G}(\text{Init}^{\overline{\cdot}})$$

Special case when $n = 0$ is truly initial:

$$\frac{P_o \in \mathcal{P}}{\Gamma, P_o, \neg P_o} \mathcal{G}(\text{Init}^{\overline{\cdot}})$$

Decomposition Rule

$$\frac{}{\Gamma, [\mathbf{A} \doteq^{\alpha} \mathbf{A}]} \mathcal{G}(Ref1)$$

- Can we restrict the type α ?
- What about $[[H_{\iota o} \top] \doteq^{\iota} [H_{\iota o} \blacksquare \neg \perp]]$?

Decomposition Rule

$$\frac{}{\Gamma, [A] \vdash A} \mathcal{G}(RefI)$$

- Can we restrict the type α ?
- What about $[[H_{\iota\sigma} \top] \doteq^{\iota} [H_{\iota\sigma} \neg \perp]]$?

Decomposition Rule

$n \geq 0 :$

$$\frac{\Gamma, [\mathbf{A}^1 \doteq \mathbf{B}^1] \quad \dots \quad \Gamma, [\mathbf{A}^n \doteq \mathbf{B}^n] \quad H_{\iota\alpha^n \dots \alpha^1} \in \mathcal{P}}{\Gamma, [[H \overline{\mathbf{A}^n}] \doteq^{\iota} [H \overline{\mathbf{B}^n}]]} \mathcal{G}(Dec)$$

Decomposition Rule

$n \geq 0 :$

$$\frac{\Gamma, [\mathbf{A}^1 \doteq \mathbf{B}^1] \quad \dots \quad \Gamma, [\mathbf{A}^n \doteq \mathbf{B}^n] \quad H_{\iota\alpha^n \dots \alpha^1} \in \mathcal{P}}{\Gamma, [[H \overline{\mathbf{A}^n}] \doteq^{\iota} [H \overline{\mathbf{B}^n}]]} \mathcal{G}(Dec)$$

Special case when $n = 0$:

$$\frac{H_{\iota} \in \mathcal{P}}{\Gamma, [H_{\iota} \doteq^{\iota} H_{\iota}]} \mathcal{G}(Dec)$$

Rules for Equality on type ι

$$\Gamma, \neg[A \doteq^{\iota} B], [C \doteq^{\iota} D]$$

Rules for Equality on type ι

$$\frac{\Gamma, [\mathbf{A} \doteq^{\iota} \mathbf{C}] \quad \Gamma, [\mathbf{B} \doteq^{\iota} \mathbf{D}]}{\Gamma, \neg[\mathbf{A} \doteq^{\iota} \mathbf{B}], [\mathbf{C} \doteq^{\iota} \mathbf{D}]} \mathcal{G}(EU_{ni}f_1)$$

Rules for Equality on type ι

$$\frac{\Gamma, [\mathbf{A} \doteq^\iota \mathbf{C}] \quad \Gamma, [\mathbf{B} \doteq^\iota \mathbf{D}]}{\Gamma, \neg[\mathbf{A} \doteq^\iota \mathbf{B}], [\mathbf{C} \doteq^\iota \mathbf{D}]} \mathcal{G}(EU\text{ni}f_1)$$

$$\frac{\Gamma, [\mathbf{A} \doteq^\iota \mathbf{D}] \quad \Gamma, [\mathbf{B} \doteq^\iota \mathbf{C}]}{\Gamma, \neg[\mathbf{A} \doteq^\iota \mathbf{B}], [\mathbf{C} \doteq^\iota \mathbf{D}]} \mathcal{G}(EU\text{ni}f_2)$$

Rules for Equality on Other Types

$$\Gamma, \neg[A =^o B]$$

Rules for Equality on Other Types

$$\frac{\Gamma, \mathbf{A}, \mathbf{B} \quad \Gamma, \neg \mathbf{A}, \neg \mathbf{B}}{\Gamma, \neg[\mathbf{A} \doteq \mathbf{B}]} \mathcal{G}(\neg \doteq)$$

Rules for Equality on Other Types

$$\frac{\Gamma, \mathbf{A}, \mathbf{B} \quad \Gamma, \neg \mathbf{A}, \neg \mathbf{B}}{\Gamma, \neg[\mathbf{A} \doteq^o \mathbf{B}]} \mathcal{G}(\neg \doteq^o)$$

$$\Gamma, \neg[\mathbf{G} \doteq^{\alpha\beta} \mathbf{H}]$$

Rules for Equality on Other Types

$$\frac{\Gamma, \mathbf{A}, \mathbf{B} \quad \Gamma, \neg \mathbf{A}, \neg \mathbf{B}}{\Gamma, \neg[\mathbf{A} \doteq^o \mathbf{B}]} \mathcal{G}(\neg \doteq^o)$$

$$\frac{\Gamma, \neg[[\mathbf{G} \mathbf{B}] \doteq^\alpha [\mathbf{H} \mathbf{B}]] \quad \mathbf{B} \in \text{cwff}_\beta(\mathcal{S})}{\Gamma, \neg[\mathbf{G} \doteq^{\alpha\beta} \mathbf{H}]} \mathcal{G}(\neg \doteq^\rightarrow, \mathcal{S})$$

Extensional Rules

$$\Gamma, [A \stackrel{\circ}{=} B]$$

Extensional Rules

Boolean Extensionality:

$$\frac{\Gamma, \neg \mathbf{A}, \mathbf{B} \quad \Gamma, \neg \mathbf{B}, \mathbf{A}}{\Gamma, [\mathbf{A} \stackrel{\circ}{=} \mathbf{B}]} \mathcal{G}(\mathfrak{b})$$

Extensional Rules

Boolean Extensionality:

$$\frac{\Gamma, \neg \mathbf{A}, \mathbf{B} \quad \Gamma, \neg \mathbf{B}, \mathbf{A}}{\Gamma, [\mathbf{A} \stackrel{\circ}{=} \mathbf{B}]} \mathcal{G}(\mathfrak{b})$$

$$\Gamma, [\mathbf{G} \stackrel{\alpha\beta}{=} \mathbf{H}]$$

Extensional Rules

Boolean Extensionality:

$$\frac{\Gamma, \neg \mathbf{A}, \mathbf{B} \quad \Gamma, \neg \mathbf{B}, \mathbf{A}}{\Gamma, [\mathbf{A} \doteq^0 \mathbf{B}]} \mathcal{G}(\mathfrak{b})$$

Functional Extensionality:

$$\frac{\Gamma, [[\mathbf{G} \mathbf{W}] \doteq^\alpha [\mathbf{H} \mathbf{W}]] \quad \mathbf{W} \in \mathcal{P}_\beta \setminus \mathbf{Params}(\Gamma, [\mathbf{G} \doteq \mathbf{H}])}{\Gamma, [\mathbf{G} \doteq^{\alpha\beta} \mathbf{H}]} \mathcal{G}(f^{\mathbf{W}})$$

Simple Extensional Proof

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

Simple Extensional Proof

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

$$[\neg A_o \vee \cdot \neg B_o \vee \cdot \neg [P_{oo} A] \vee [P B]]$$

Simple Extensional Proof

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

$$\frac{\neg A, [\neg B \vee \cdot \neg[P A] \vee [P B]]}{[\neg A_o \vee \cdot \neg B_o \vee \cdot \neg[P_{oo} A] \vee [P B]]} \mathcal{G}(\vee)$$

Simple Extensional Proof

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

$$\frac{\neg A, \neg B, [\neg[P A] \vee [P B]]}{\neg A, [\neg B \vee \cdot \neg[P A] \vee [P B]]} \mathcal{G}(\vee)$$
$$\frac{\neg A, [\neg B \vee \cdot \neg[P A] \vee [P B]]}{[\neg A_o \vee \cdot \neg B_o \vee \cdot \neg[P_{oo} A] \vee [P B]]} \mathcal{G}(\vee)$$

Simple Extensional Proof

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

$$\frac{\frac{\frac{\neg A, \neg B, \neg[PA], [PB]}{\neg A, \neg B, [\neg[PA] \vee [PB]]} \mathcal{G}(\vee)}{\neg A, [\neg B \vee \cdot \neg[PA] \vee [PB]]} \mathcal{G}(\vee)}{[\neg A_o \vee \cdot \neg B_o \vee \cdot \neg[P_{oo} A] \vee [PB]]} \mathcal{G}(\vee)$$

Simple Extensional Proof

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

$$\frac{\frac{\frac{\neg A, \neg B, [B \overset{o}{=} A]}{\neg A, \neg B, \neg [P A], [P B]} \mathcal{G}(Init^{\overline{=}})}{\neg A, \neg B, [\neg [P A] \vee [P B]]} \mathcal{G}(\vee)}{\neg A, [\neg B \vee \cdot \neg [P A] \vee [P B]]} \mathcal{G}(\vee)}{\neg A_o \vee \cdot \neg B_o \vee \cdot \neg [P_{oo} A] \vee [P B]} \mathcal{G}(\vee)$$

Simple Extensional Proof

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

$$\begin{array}{c}
 \frac{\frac{\frac{\frac{\frac{\frac{\neg A, \neg B, \neg B, A}{\neg A, \neg B, \neg A, B}}{\neg A, \neg B, [B =^o A]}{\neg A, \neg B, \neg [P A], [P B]}{\neg A, \neg B, [\neg [P A] \vee [P B]}{\neg A, [\neg B \vee \cdot \neg [P A] \vee [P B]}]}{\neg A, \neg B, \neg [P A], [P B]} \mathcal{G}(Init^=)}{\neg A, \neg B, [\neg [P A] \vee [P B]} \mathcal{G}(\vee)}{\neg A, [\neg B \vee \cdot \neg [P A] \vee [P B]} \mathcal{G}(\vee)}{\neg A_o \vee \cdot \neg B_o \vee \cdot \neg [P_{oo} A] \vee [P B]} \mathcal{G}(\vee)} \mathcal{G}(b)
 \end{array}$$

Simple Extensional Proof

$$[A_o \supset \cdot B_o \supset \cdot [P_{oo} A_o] \supset [P_{oo} B_o]]$$

$$\begin{array}{c}
 \frac{}{\neg A, \neg B, \neg B, A} \mathcal{G}(Init^{\bar{=}}) \quad \frac{}{\neg A, \neg B, \neg A, B} \mathcal{G}(Init^{\bar{=}}) \\
 \hline
 \neg A, \neg B, \neg B, A \quad \neg A, \neg B, \neg A, B \\
 \hline
 \neg A, \neg B, [B \stackrel{o}{=} A] \\
 \hline
 \neg A, \neg B, \neg[P A], [P B] \quad \mathcal{G}(Init^{\bar{=}}) \\
 \hline
 \neg A, \neg B, [\neg[P A] \vee [P B]] \quad \mathcal{G}(\vee) \\
 \hline
 \neg A, \neg B, [\neg B \vee \cdot \neg[P A] \vee [P B]] \quad \mathcal{G}(\vee) \\
 \hline
 \neg A_o \vee \cdot \neg B_o \vee \cdot \neg[P_{oo} A_o] \vee [P_{oo} B_o] \quad \mathcal{G}(\vee)
 \end{array}$$

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})
- Derived Rules for Abbreviations:

$$\frac{\Gamma, \neg M, N}{\Gamma, [M \supset N]} \mathcal{G}(\supset)$$

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})
- Derived Rules for Abbreviations:

$$\frac{\Gamma, \neg M, N}{\Gamma, [M \supset N]} \mathcal{G}(\supset)$$

$$\frac{\Gamma, M \quad \Gamma, N}{\Gamma, [M \wedge N]} \mathcal{G}(\wedge)$$

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})
- Derived Rules for Abbreviations:

$$\frac{\Gamma, \neg M, N}{\Gamma, [M \supset N]} \mathcal{G}(\supset)$$

$$\frac{\Gamma, M \quad \Gamma, N}{\Gamma, [M \wedge N]} \mathcal{G}(\wedge)$$

$$\frac{\Gamma, [A/x]M \quad A \in \text{cwff}_\alpha(\mathcal{S})}{\Gamma, [\exists x_\alpha M]} \mathcal{G}(\exists, \mathcal{S})$$

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})
- Sound and Complete w.r.t. \mathcal{S} -Models (Chaps. 2-5)

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})
- Sound and Complete w.r.t. \mathcal{S} -Models (Chaps. 2-5)
- Admissible Rules (by Completeness):

$$\frac{}{\Gamma, \mathbf{M}, \neg \mathbf{M}} \mathcal{G}(GInit)$$

$$\frac{}{\Gamma, [\mathbf{A} \stackrel{\alpha}{=} \mathbf{A}]} \mathcal{G}(Ref1)$$

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})
- Sound and Complete w.r.t. \mathcal{S} -Models (Chaps. 2-5)
- Admissible Rules (by Completeness):

$$\frac{}{\Gamma, \mathbf{M}, \neg \mathbf{M}} \mathcal{G}(GInit) \qquad \frac{}{\Gamma, [\mathbf{A} \stackrel{\alpha}{=} \mathbf{A}]} \mathcal{G}(Refl)$$

$$\frac{\Gamma, \mathbf{M} \quad \Gamma, \neg \mathbf{M}}{\Gamma} \mathcal{G}(Cut)$$

Cut-Elimination!

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})
- Sound and Complete w.r.t. \mathcal{S} -Models (Chaps. 2-5)
- Conservation and Independence Results (Chap. 6)

Extensional Sequent Calculus

- Rules Define a Sequent Calculus (Relative to \mathcal{S})
- Sound and Complete w.r.t. \mathcal{S} -Models (Chaps. 2-5)
- Conservation and Independence Results (Chap. 6)
- Enough to Consider Signature

$$\{\neg, \vee, =^{\iota}\} \cup \{\Pi^{\alpha} \mid \alpha \text{ not a propositional type}\}$$

Extensional Search

- Need a Version of Expansion Proofs for Extensional Proofs

Extensional Search

- Need a Version of Expansion Proofs for Extensional Proofs
- Generalize Expansion Trees to Extensional Expansion Dags

Extensional Search

- Need a Version of Expansion Proofs for Extensional Proofs
- Generalize Expansion Trees to Extensional Expansion Dags
- Need a Method of Searching for Extensional Expansion Proofs

Expansion Proofs and Sequent Proofs

Reconsider: $[[\forall x_t [P_{ot} x]] \supset \cdot [P A_t] \wedge [P B_t]]$

Expansion Proofs and Sequent Proofs

Reconsider: $[[\forall x_i [P_{oi} x]] \supset \cdot [P A_i] \wedge [P B_i]]$

- The proof requires no extensionality.

Expansion Proofs and Sequent Proofs

Reconsider: $[[\forall x_l [P_{o_l} x]] \supset \cdot [P A_l] \wedge [P B_l]]$

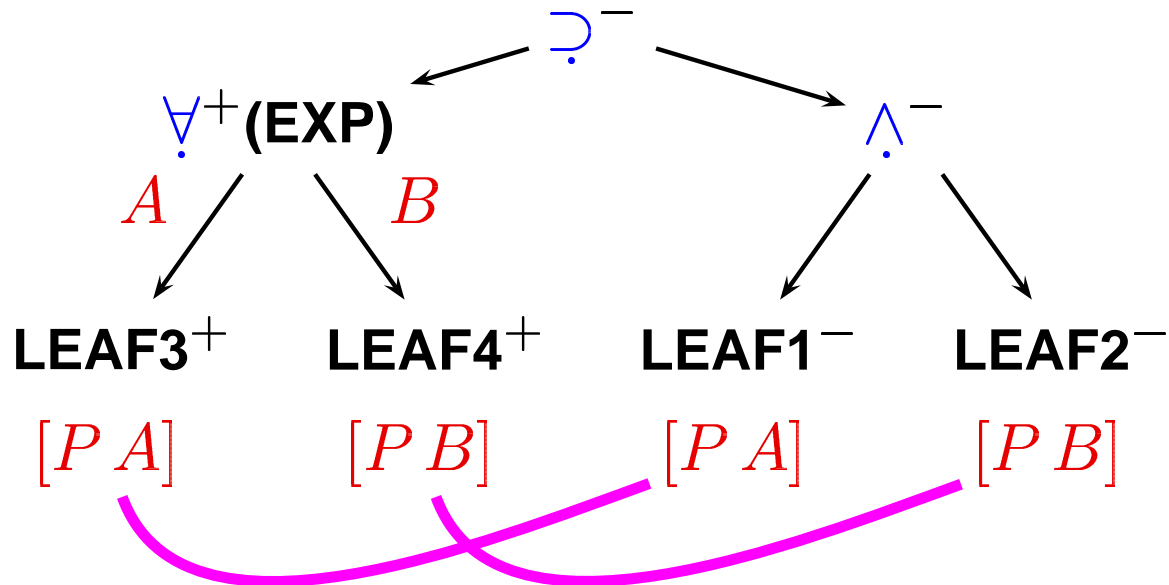
- The proof requires no extensionality.
- Expansion Proof Corresponds to Sequent Calculus Derivation

Expansion Proofs and Sequent Proofs

$$\frac{\frac{\frac{}{\neg[P_{ol} A], [P A_l]} \mathcal{G}(GInit)}{\neg \forall x_l [P_{ol} x], [P A_l]} \mathcal{G}(\neg \forall)}{\frac{\frac{}{\neg[P_{ol} B], [P B_l]} \mathcal{G}(GInit)}{\neg \forall x_l [P_{ol} x], [P B_l]} \mathcal{G}(\neg \forall)}{\frac{\neg \forall x_l [P_{ol} x], [[P A_l] \wedge [P B_l]]}{[[\forall x_l [P_{ol} x]] \supset \blacksquare [P A_l] \wedge [P B_l]]} \mathcal{G}(\supset)} \mathcal{G}(\wedge)}$$

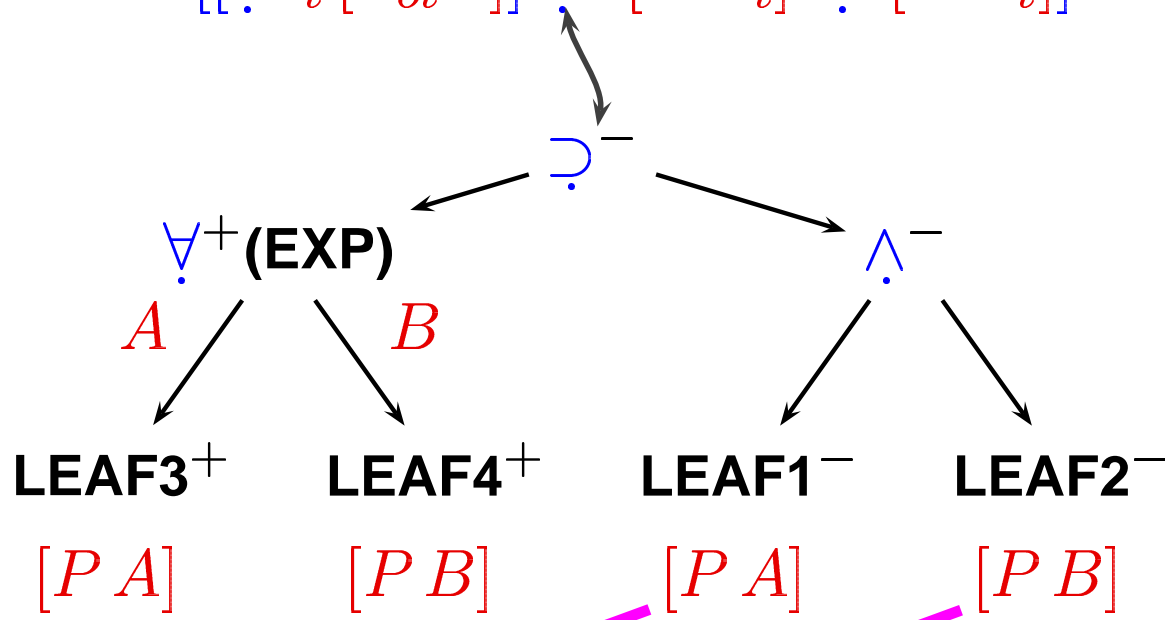
Expansion Proofs and Sequent Proofs

$$\begin{array}{c}
 \frac{}{\neg[P_{ol} A], [P A_l]} \mathcal{G}(GInit) \quad \frac{}{\neg[P_{ol} B], [P B_l]} \mathcal{G}(GInit) \\
 \frac{}{\neg \forall x_l [P_{ol} x], [P A_l]} \mathcal{G}(\neg \forall) \quad \frac{}{\neg \forall x_l [P_{ol} x], [P B_l]} \mathcal{G}(\neg \forall) \\
 \hline
 \frac{}{\neg \forall x_l [P_{ol} x], [[P A_l] \wedge [P B_l]]} \mathcal{G}(\wedge) \\
 \hline
 \frac{}{[[\forall x_l [P_{ol} x]] \supset [P A_l] \wedge [P B_l]]} \mathcal{G}(\supset)
 \end{array}$$

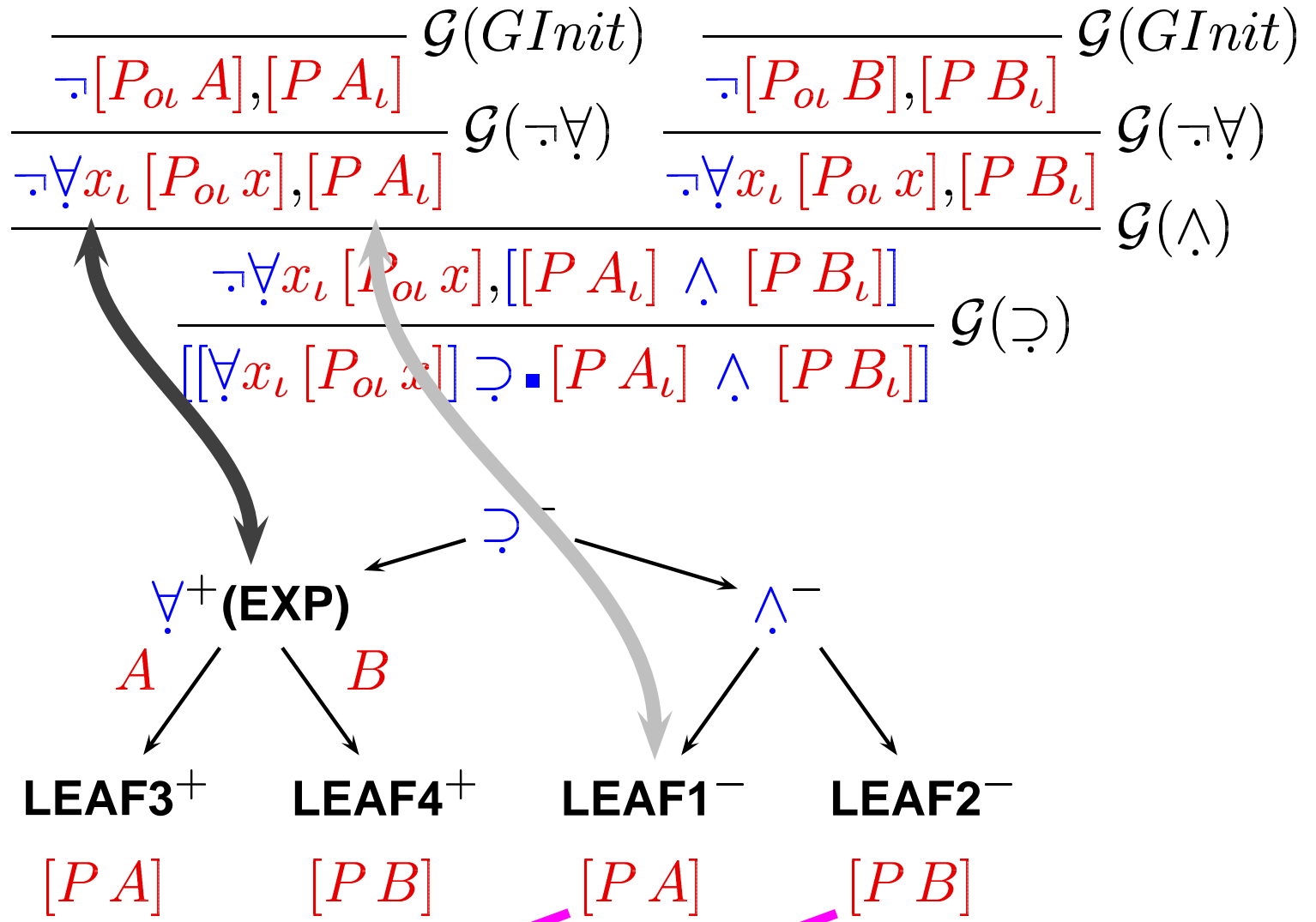


Expansion Proofs and Sequent Proofs

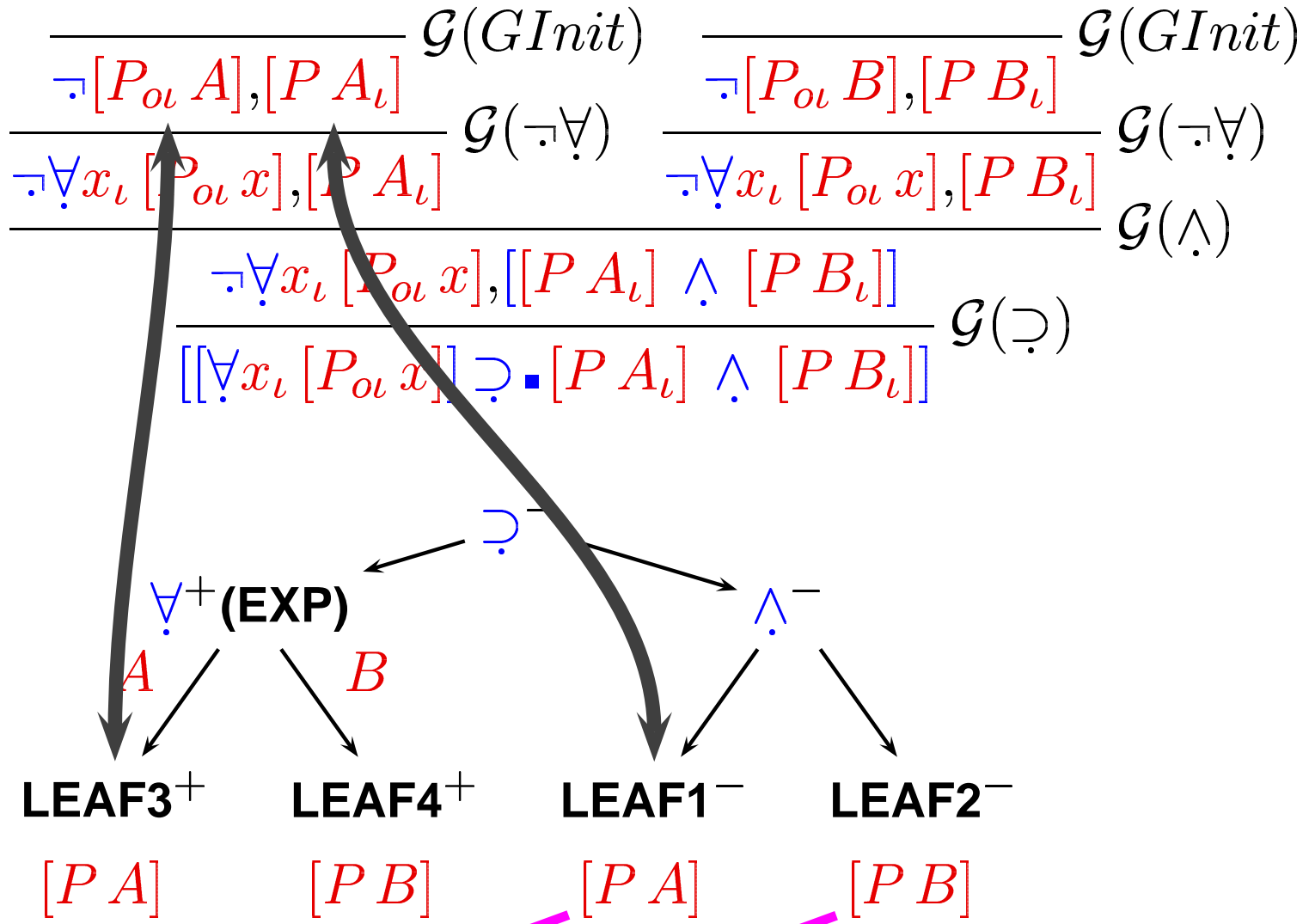
$$\begin{array}{c}
 \frac{}{\neg[P_{ol} A], [P A_l]} \mathcal{G}(GInit) \quad \frac{}{\neg[P_{ol} B], [P B_l]} \mathcal{G}(GInit) \\
 \frac{}{\neg \forall x_l [P_{ol} x], [P A_l]} \mathcal{G}(\neg \forall) \quad \frac{}{\neg \forall x_l [P_{ol} x], [P B_l]} \mathcal{G}(\neg \forall) \\
 \hline
 \frac{}{\neg \forall x_l [P_{ol} x], [[P A_l] \wedge [P B_l]]} \mathcal{G}(\wedge) \\
 \hline
 \frac{}{\neg \forall x_l [P_{ol} x], [[P A_l] \wedge [P B_l]]} \mathcal{G}(\supset) \\
 \hline
 [[\forall x_l [P_{ol} x]] \supset \cdot [P A_l] \wedge [P B_l]]
 \end{array}$$



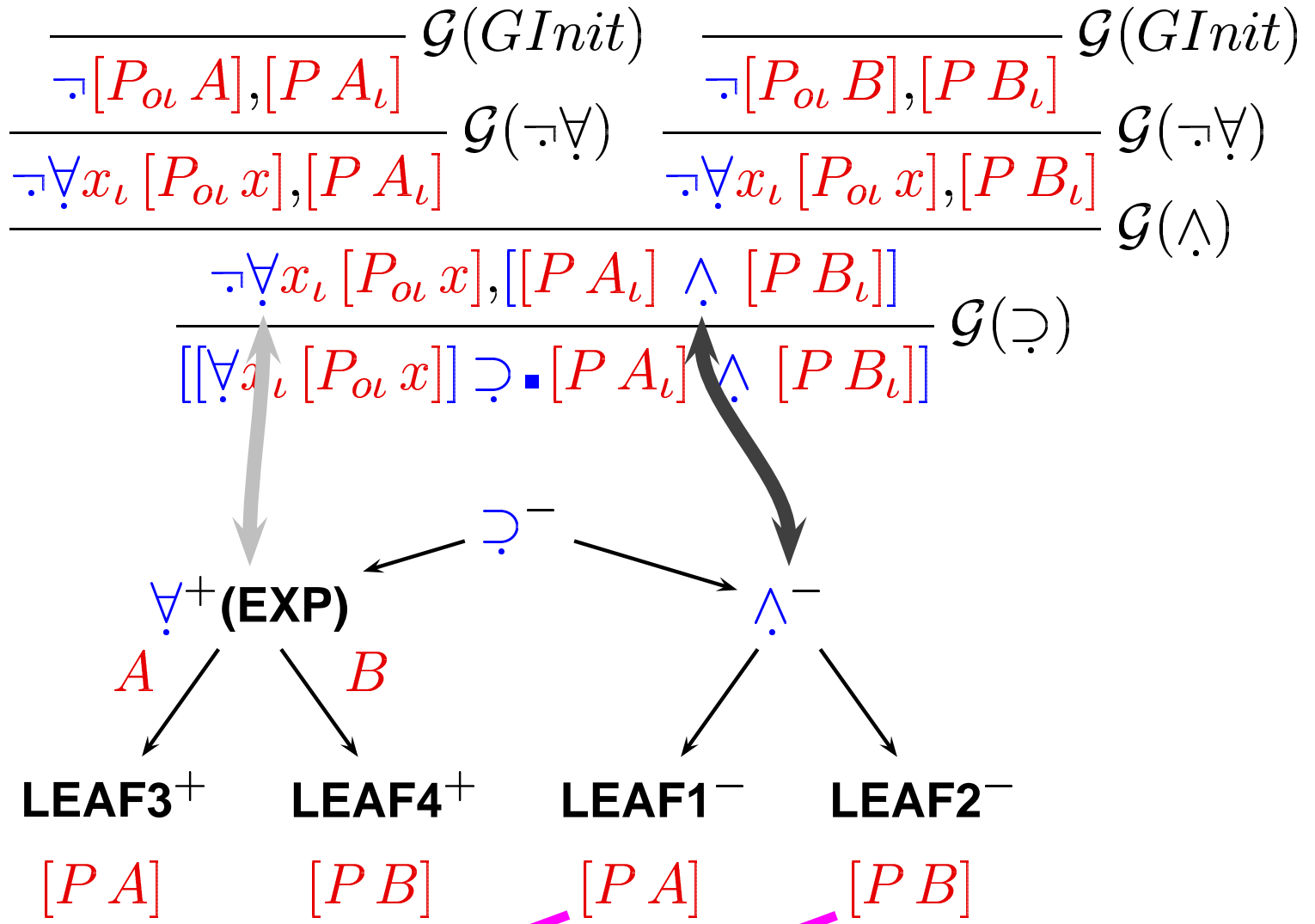
Expansion Proofs and Sequent Proofs



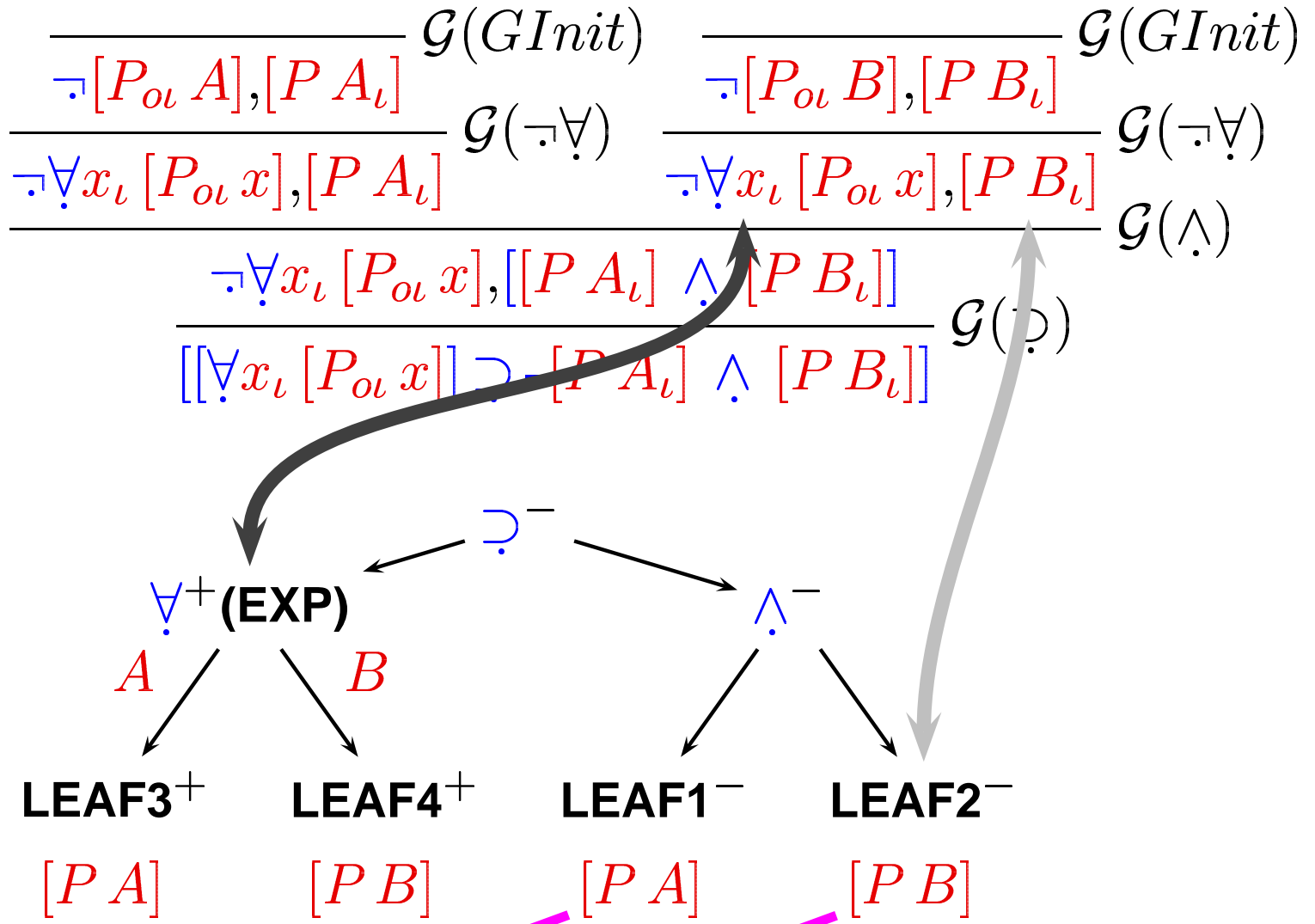
Expansion Proofs and Sequent Proofs



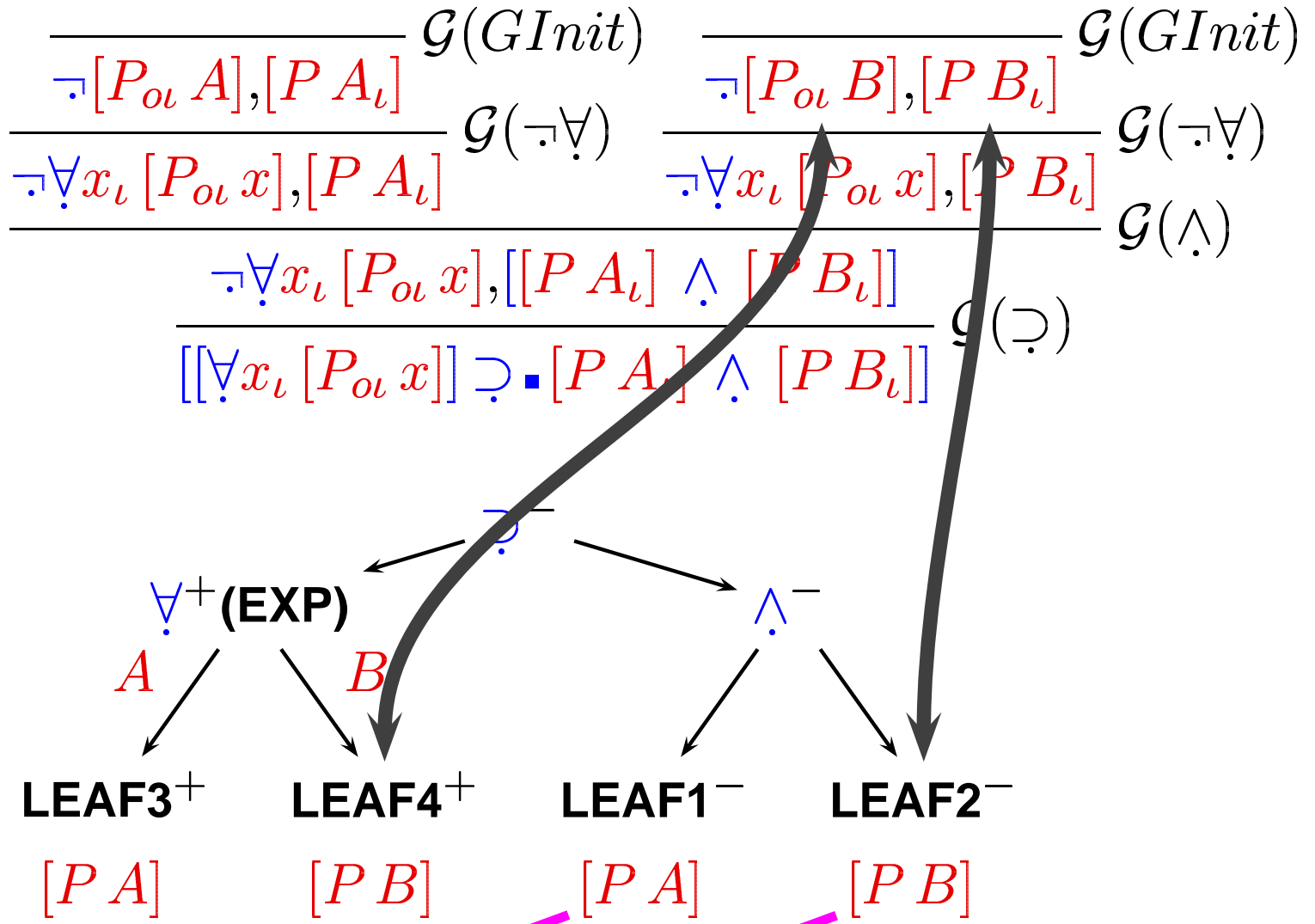
Expansion Proofs and Sequent Proofs



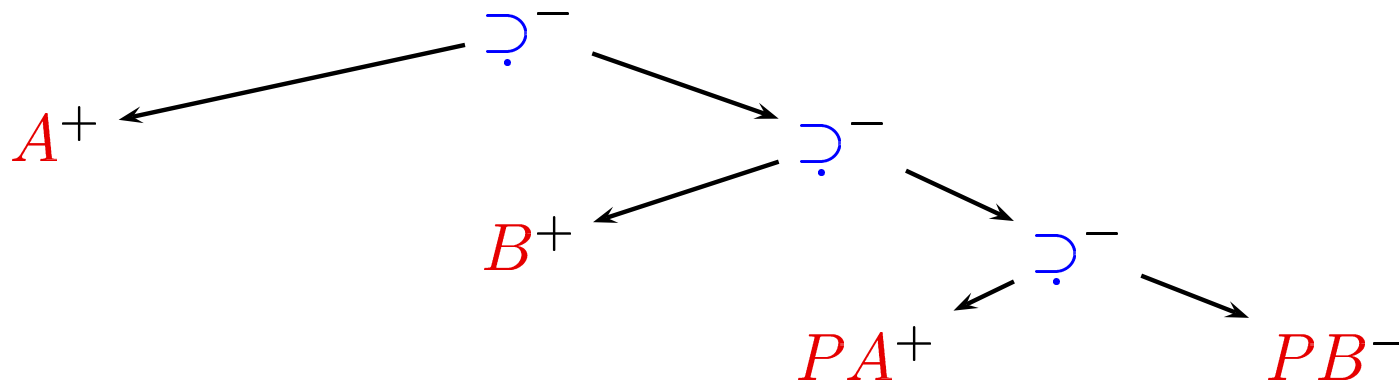
Expansion Proofs and Sequent Proofs



Expansion Proofs and Sequent Proofs

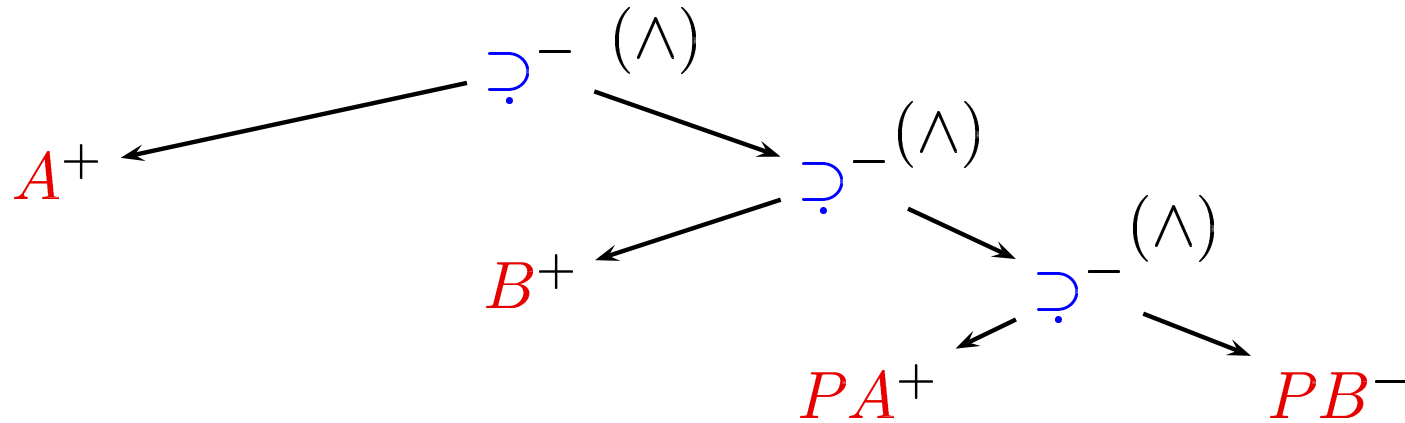


Extensional Expansion Dags



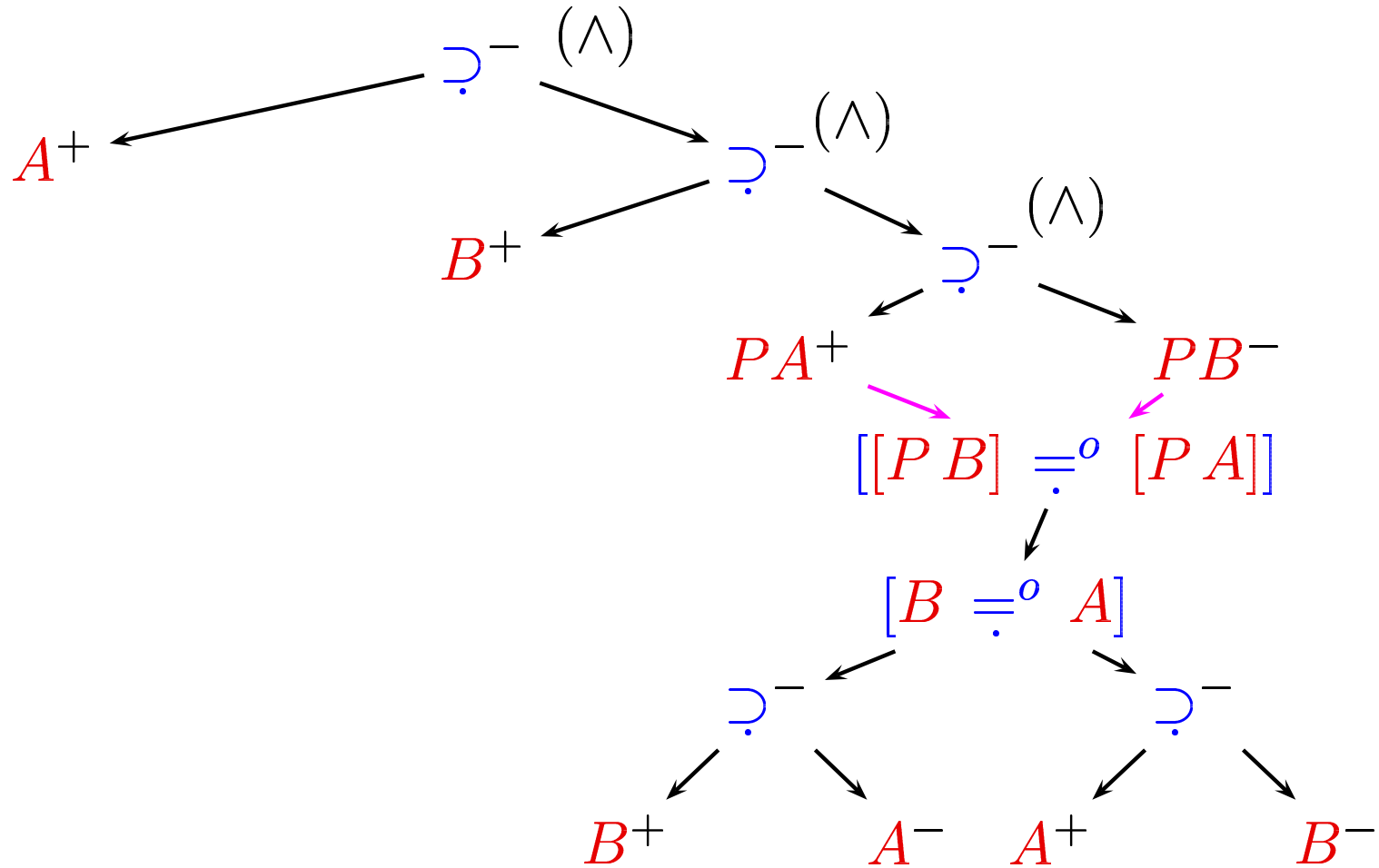
Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$

Extensional Expansion Dags

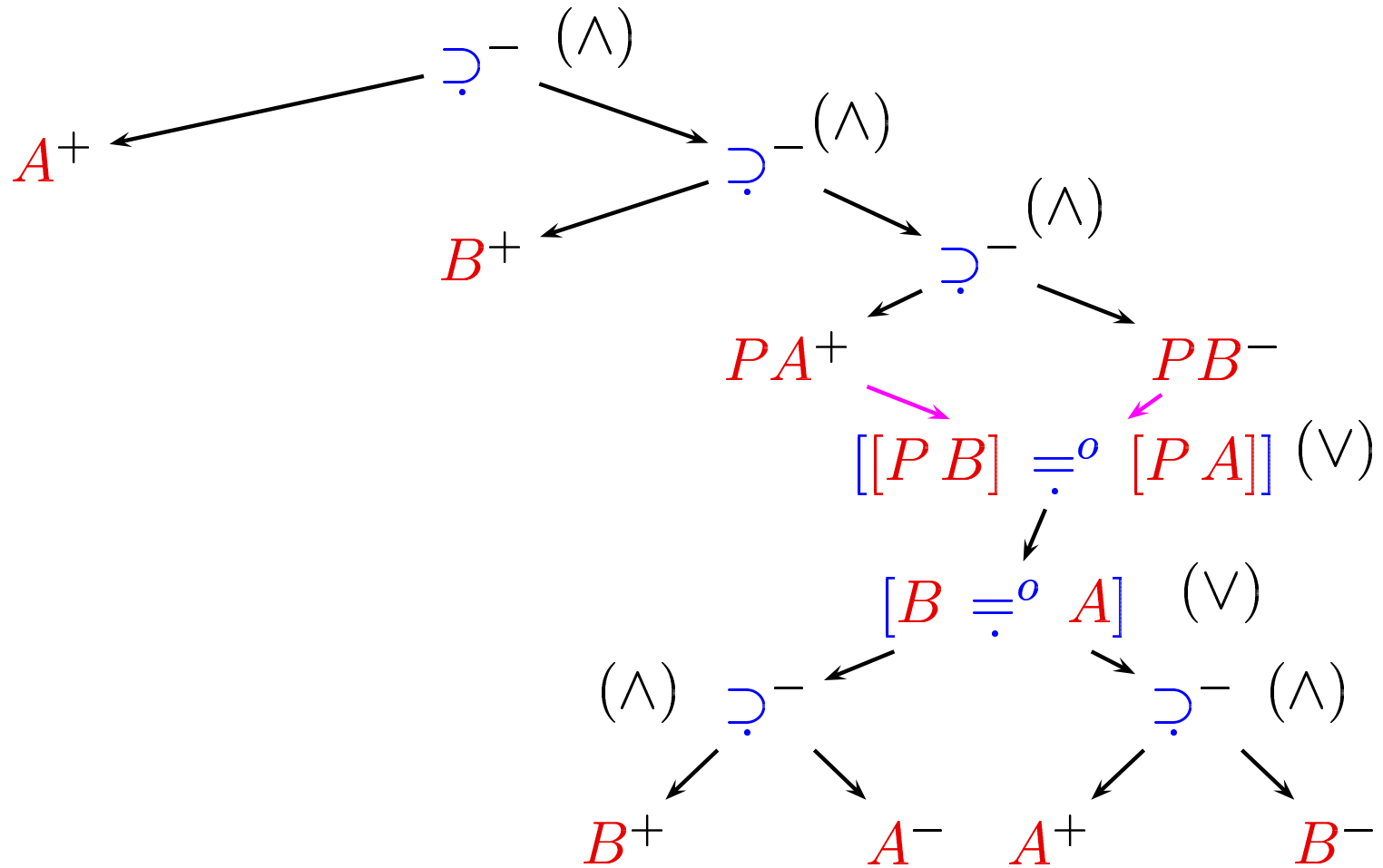


$$\begin{bmatrix} A^+ \\ B^+ \\ [PA]^+ \\ [PB]^- \end{bmatrix}$$

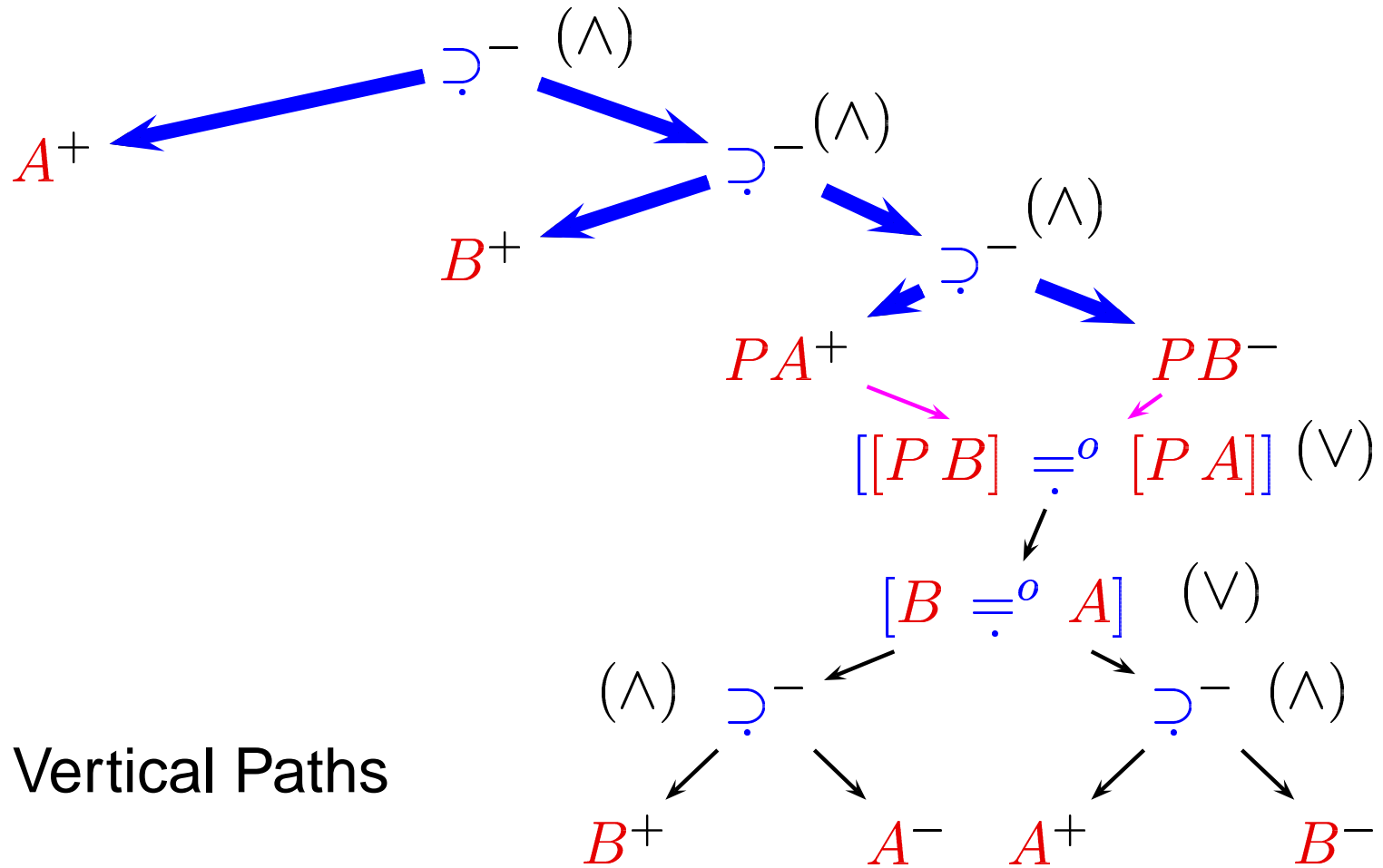
Extensional Expansion Dags



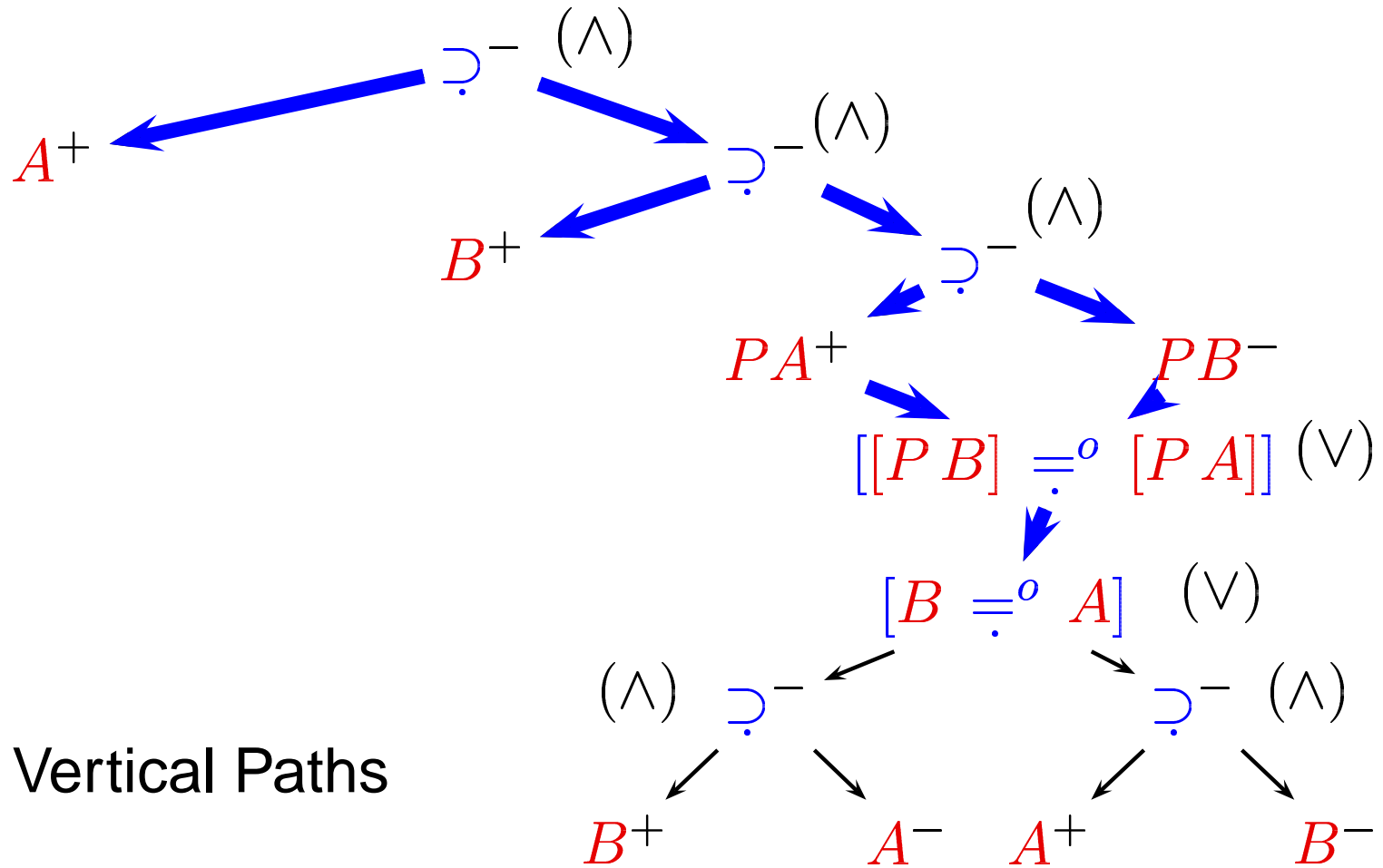
Extensional Expansion Dags



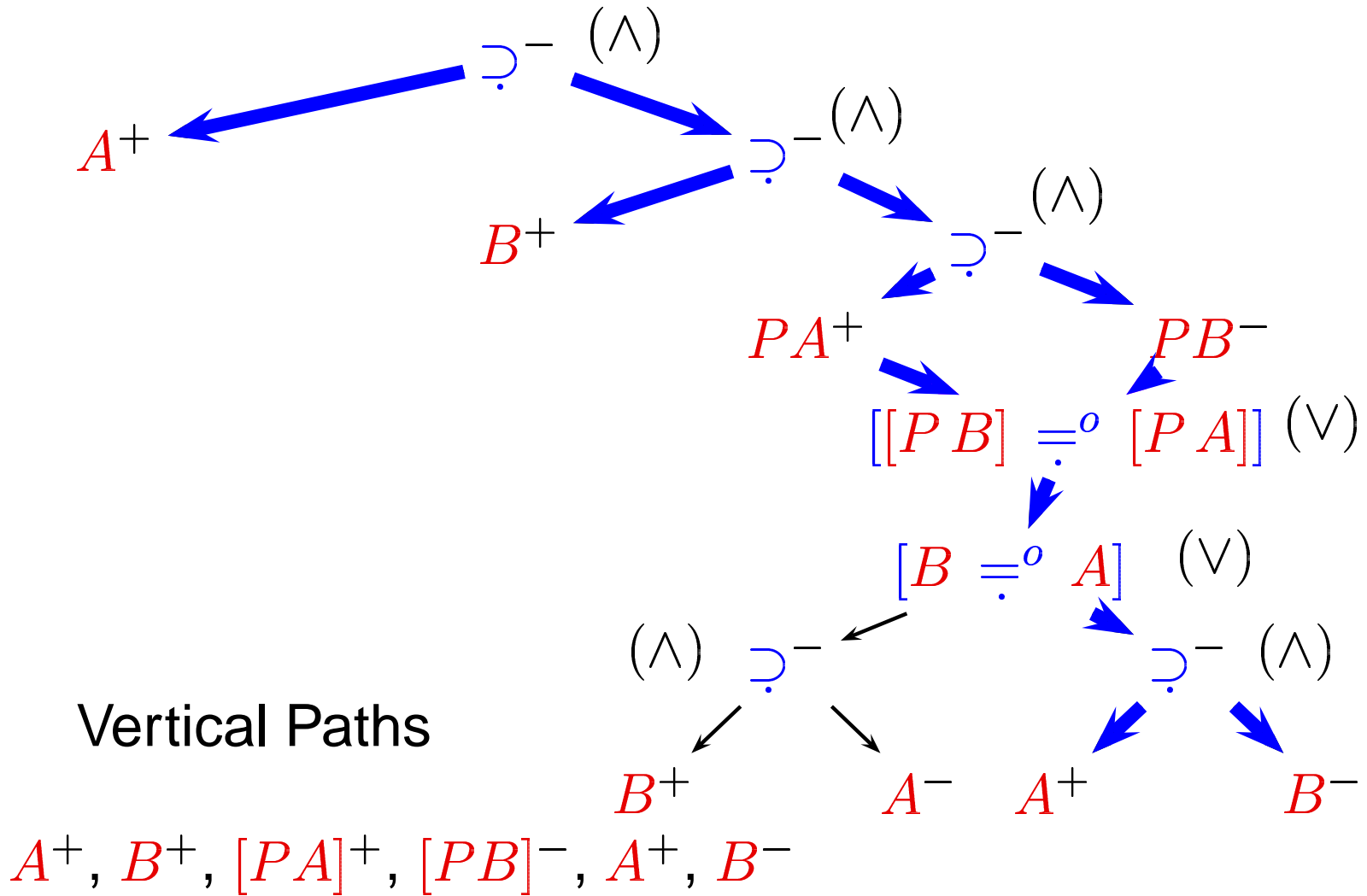
Extensional Expansion Dags



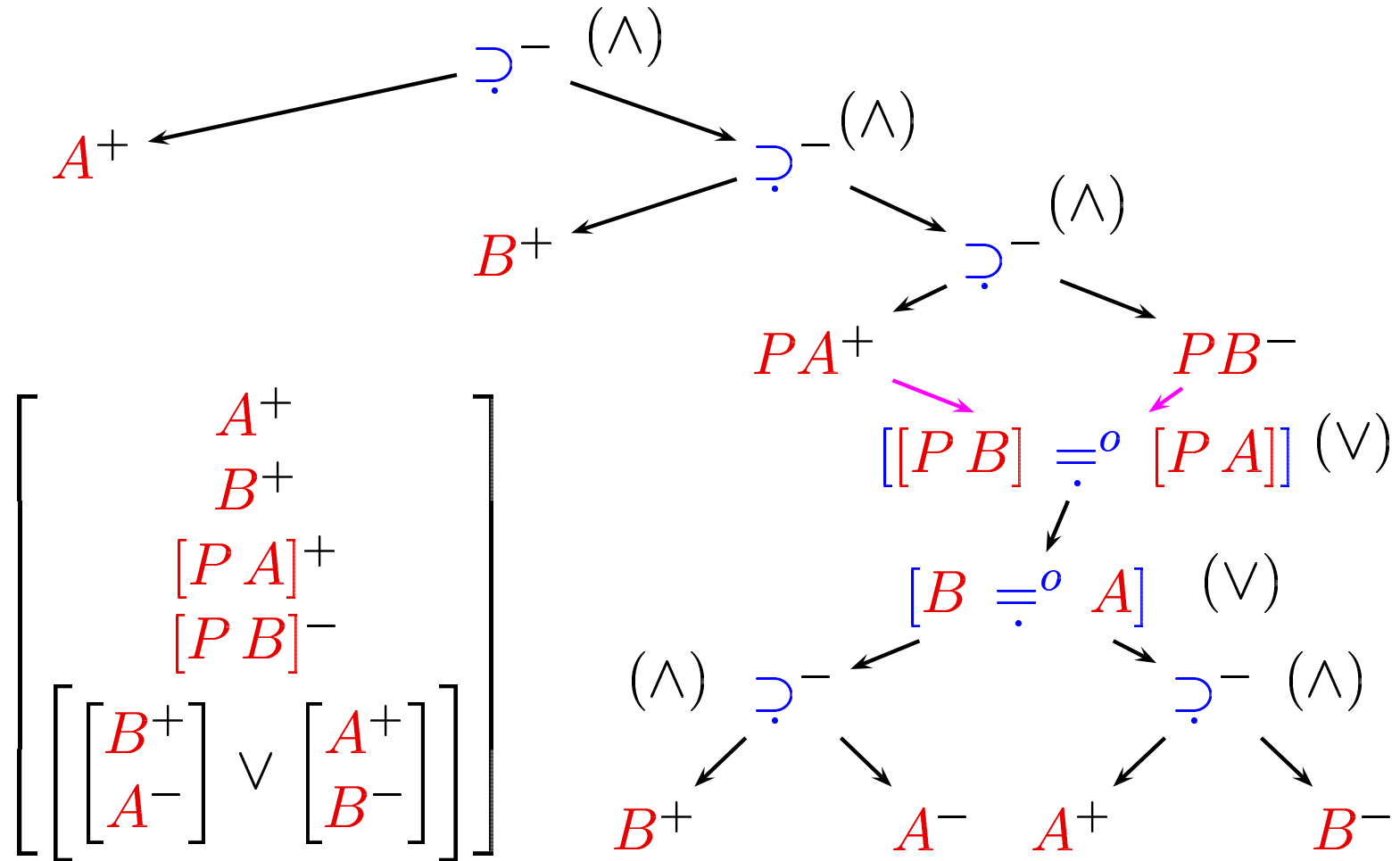
Extensional Expansion Dags



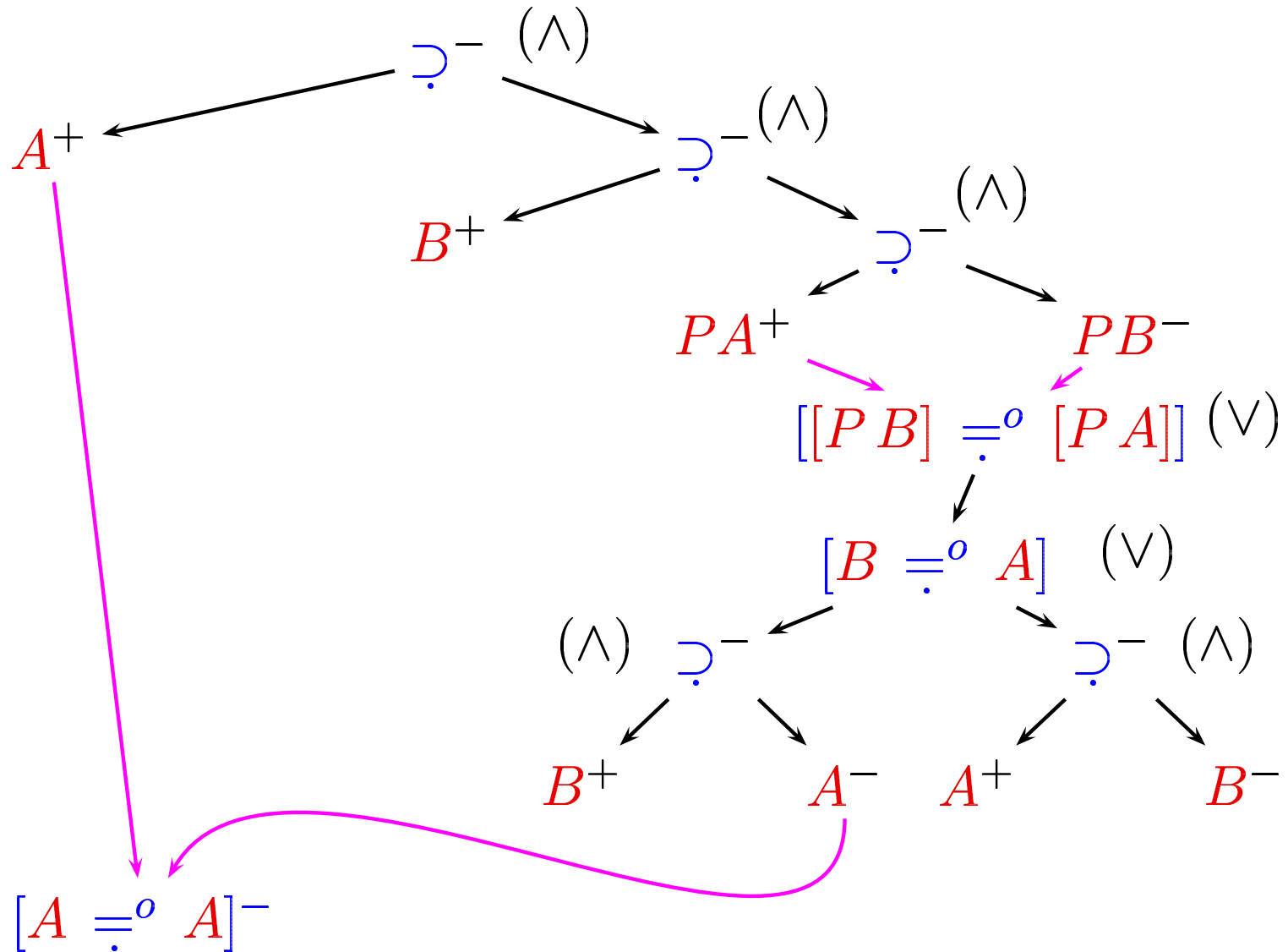
Extensional Expansion Dags



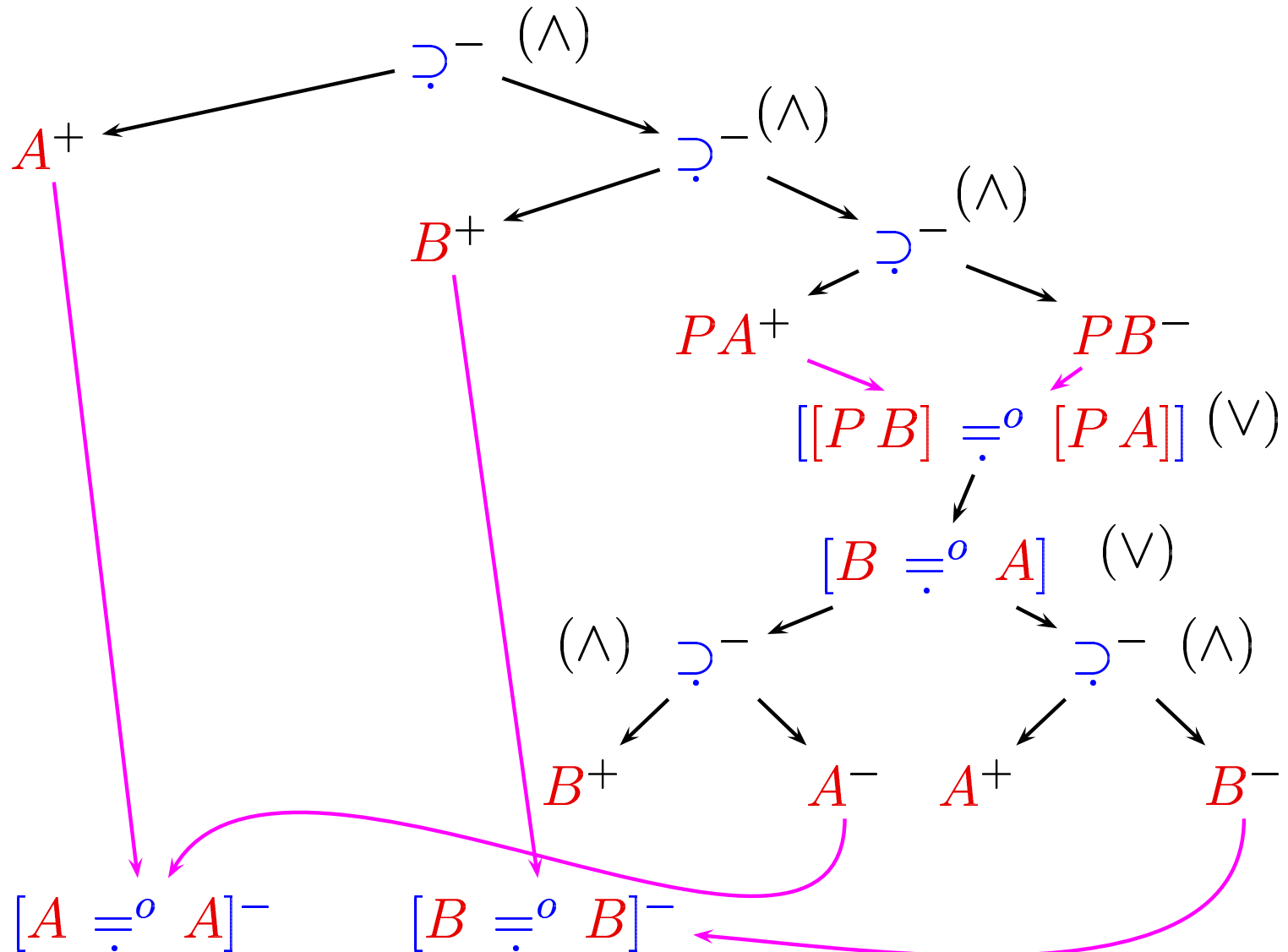
Extensional Expansion Dags



Extensional Expansion Dags



Extensional Expansion Dags



Extensional Expansion Dags

Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:

Extensional Expansion Dags


Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:

Full Derivation:

$$\begin{array}{c}
 \frac{}{\neg A, \neg B, \neg B, A} \mathcal{G}(Init^{\bar{=}}) \quad \frac{}{\neg A, \neg B, \neg A, B} \mathcal{G}(Init^{\bar{=}})}{\quad} \mathcal{G}(b) \\
 \frac{}{\neg A, \neg B, [B \stackrel{o}{=} A]} \mathcal{G}(Init^{\bar{=}})}{\quad} \mathcal{G}(\supset) \\
 \frac{}{\neg A, \neg B, \neg[P A], [P B]} \mathcal{G}(\supset)}{\quad} \mathcal{G}(\supset) \\
 \frac{}{\neg A, \neg B, [[P A] \supset [P B]]} \mathcal{G}(\supset)}{\quad} \mathcal{G}(\supset) \\
 \frac{}{\neg A, [B \supset \cdot [P A] \supset [P B]]} \mathcal{G}(\supset)}{\quad} \mathcal{G}(\supset) \\
 \frac{}{[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]} \mathcal{G}(\supset)
 \end{array}$$

Extensional Expansion Dags

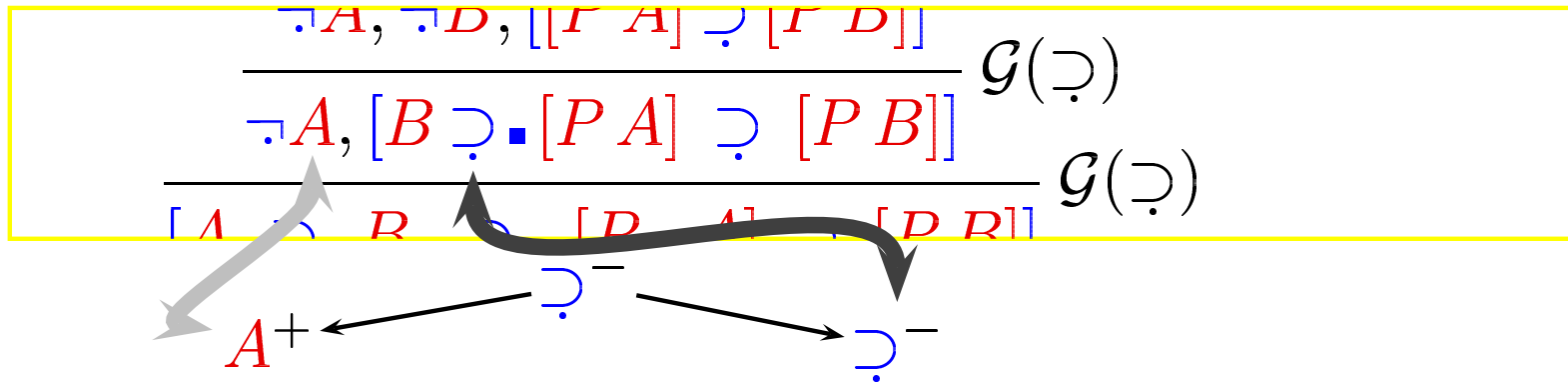
Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:

$$\frac{\supset A, [B \supset \cdot [P A] \supset [P B]]}{[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]} \mathcal{G}(\supset)$$


\supset^-

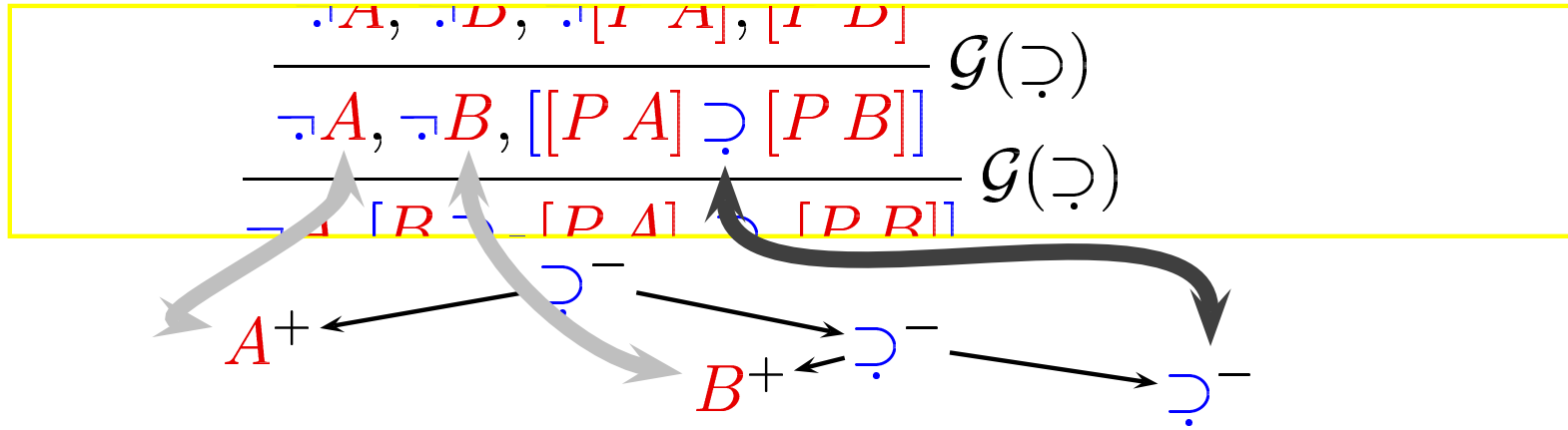
Extensional Expansion Dags

Reconsider $[A_0 \supset \cdot B_0 \supset \cdot [P_{00} A] \supset [P B]]$:



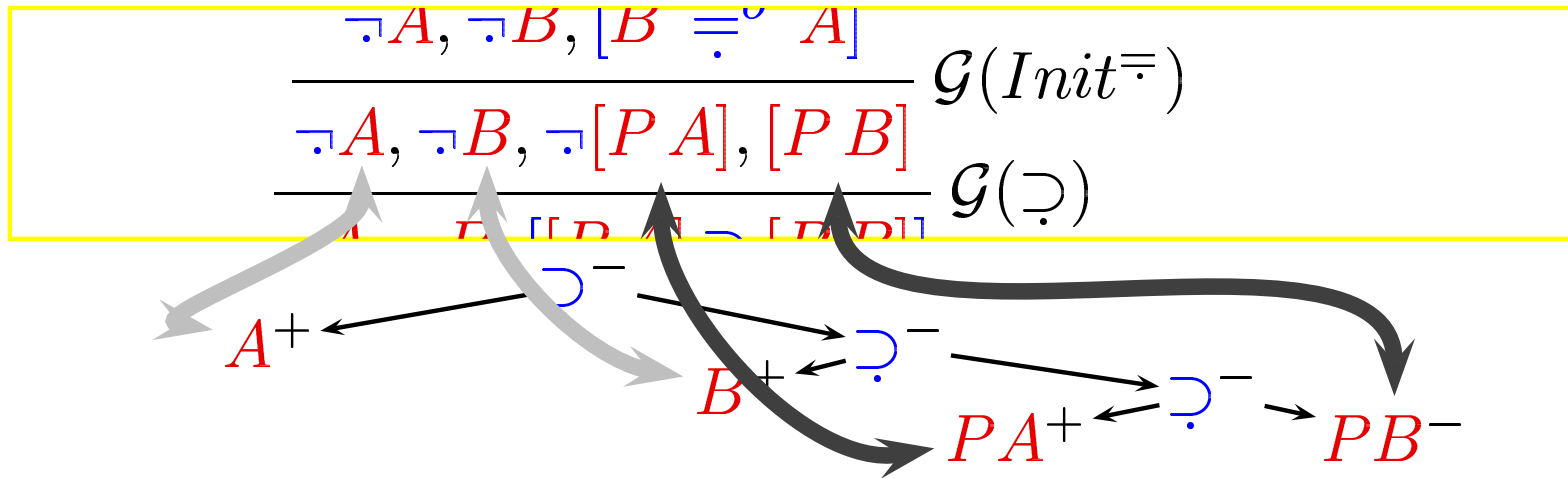
Extensional Expansion Dags

Reconsider $[A_0 \supset \cdot B_0 \supset \cdot [P_{00} A] \supset [P B]]$:



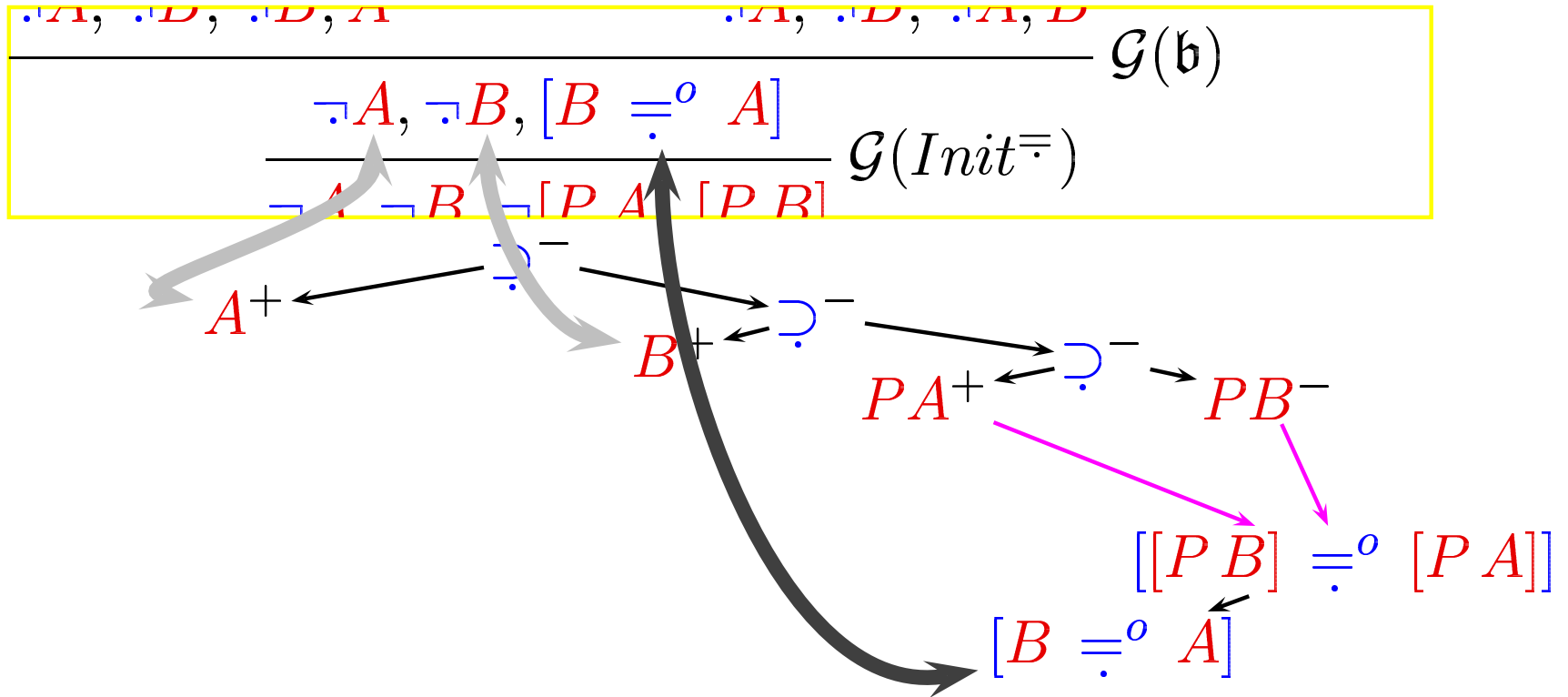
Extensional Expansion Dags

Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:



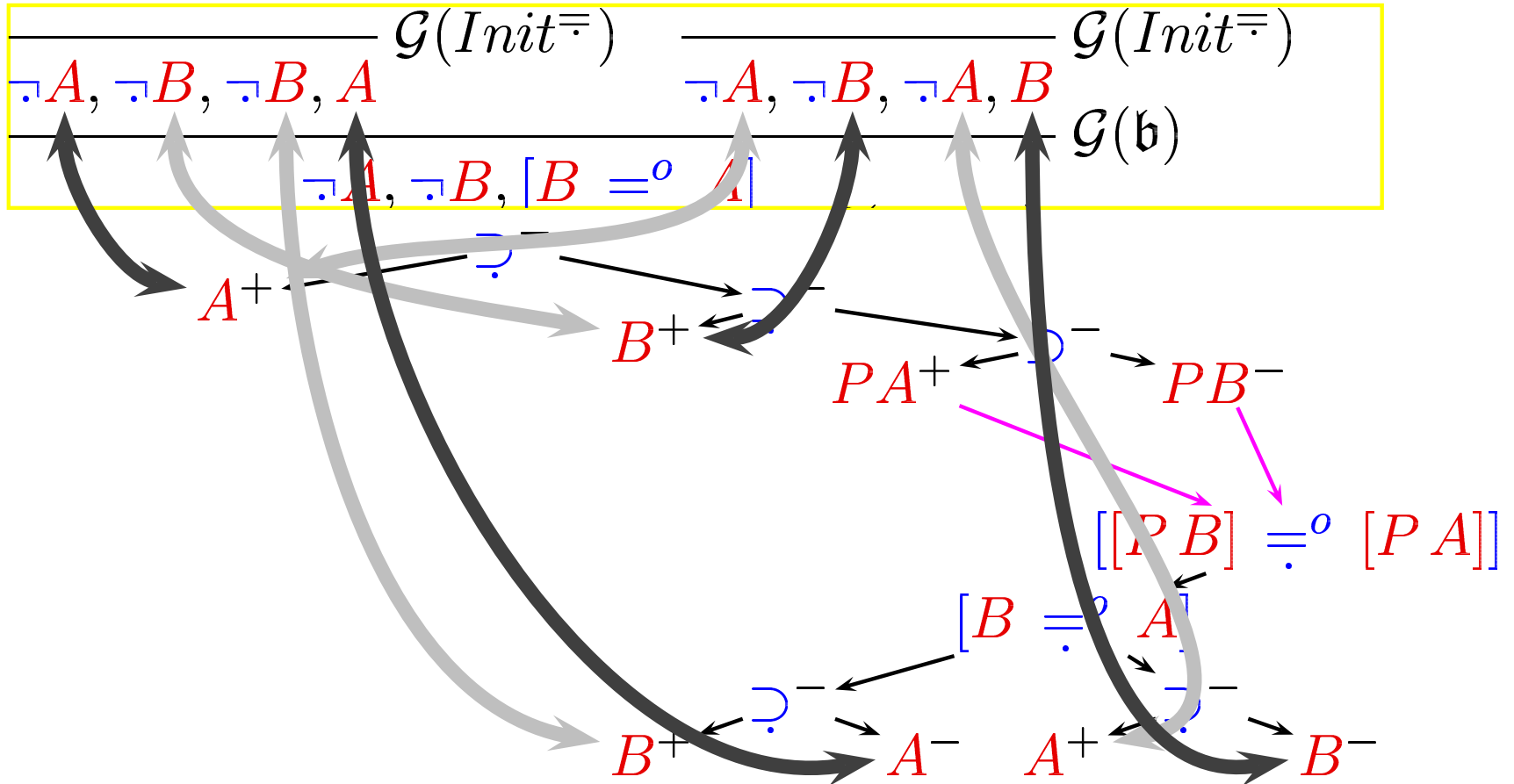
Extensional Expansion Dags

Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:



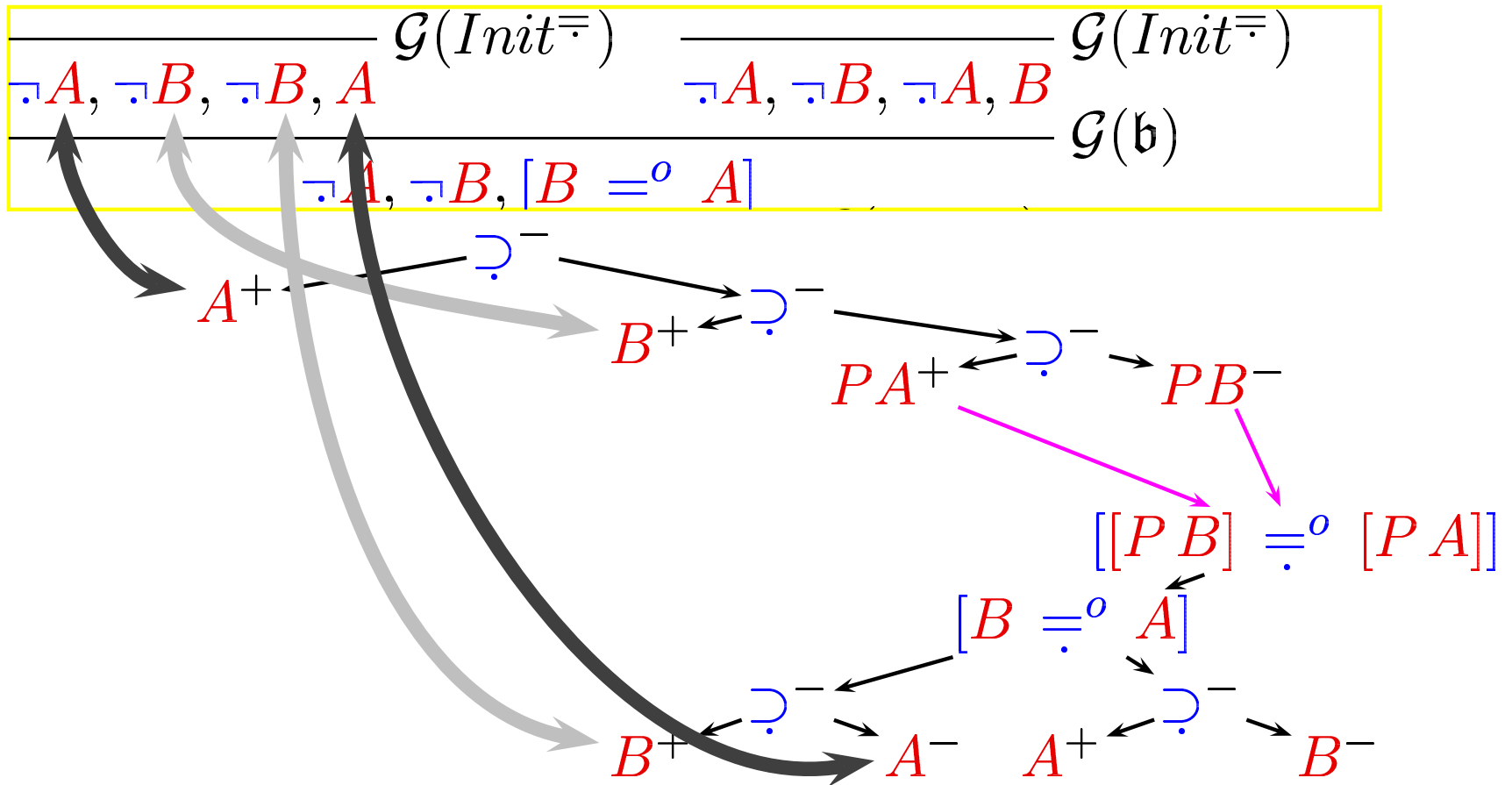
Extensional Expansion Dags

Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:



Extensional Expansion Dags

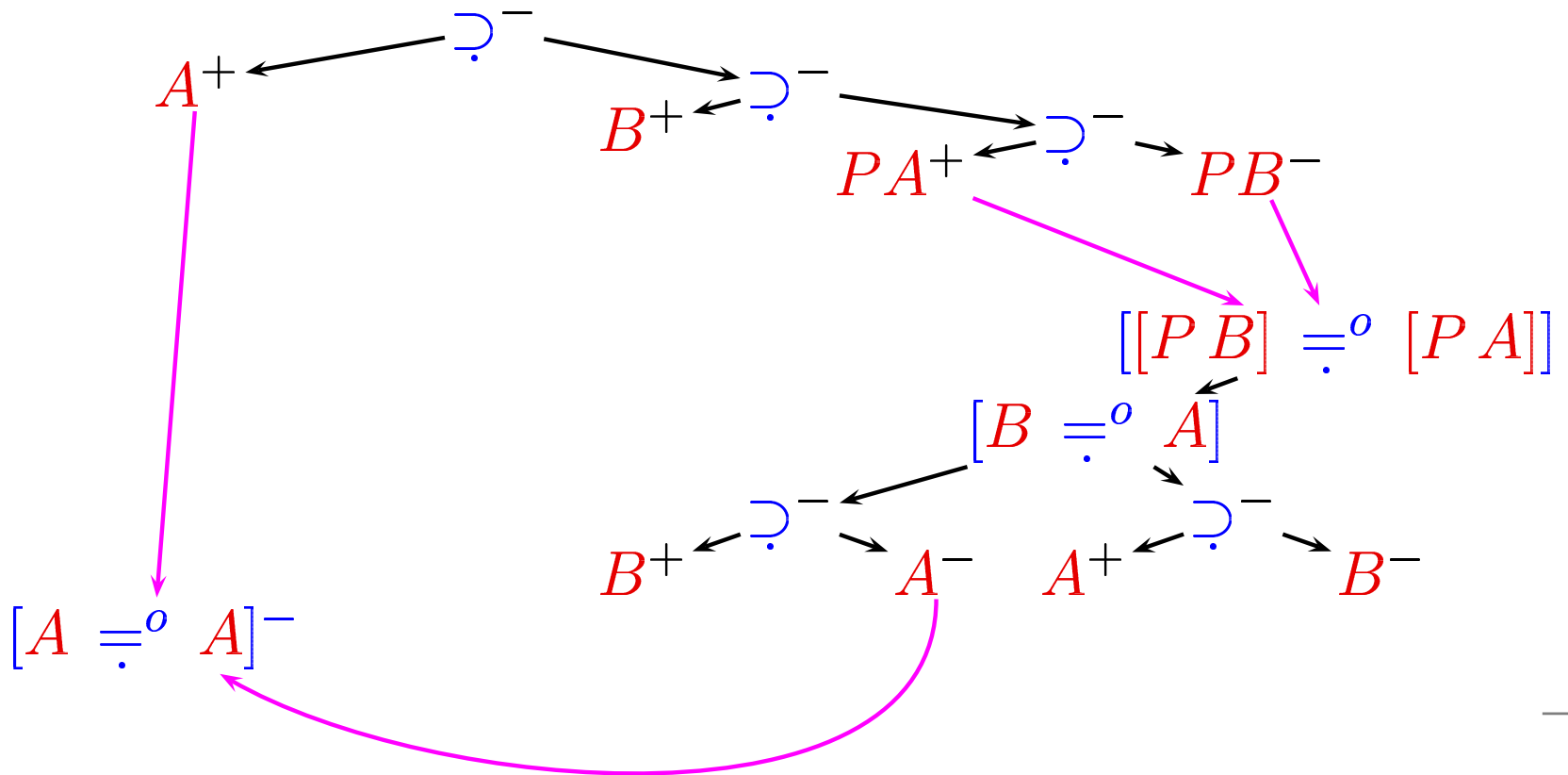
Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:



Extensional Expansion Dags

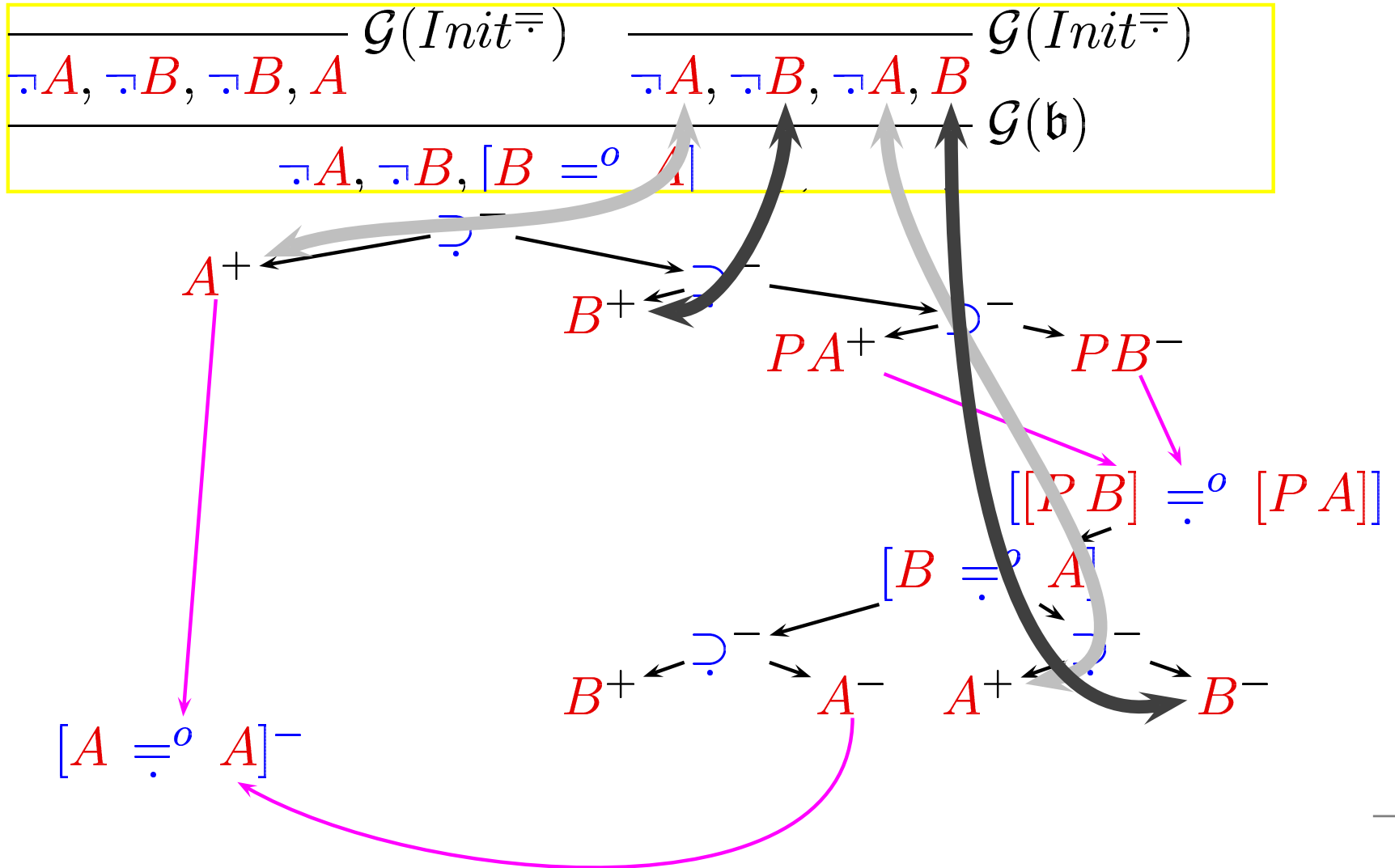
Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:

$\frac{}{\neg A, \neg B, \neg B, A} \mathcal{G}(Init^{\bar{\cdot}})$	$\frac{}{\neg A, \neg B, \neg A, B} \mathcal{G}(Init^{\bar{\cdot}})$
$\frac{}{\neg A, \neg B, [B =^o A]} \mathcal{G}(b)$	



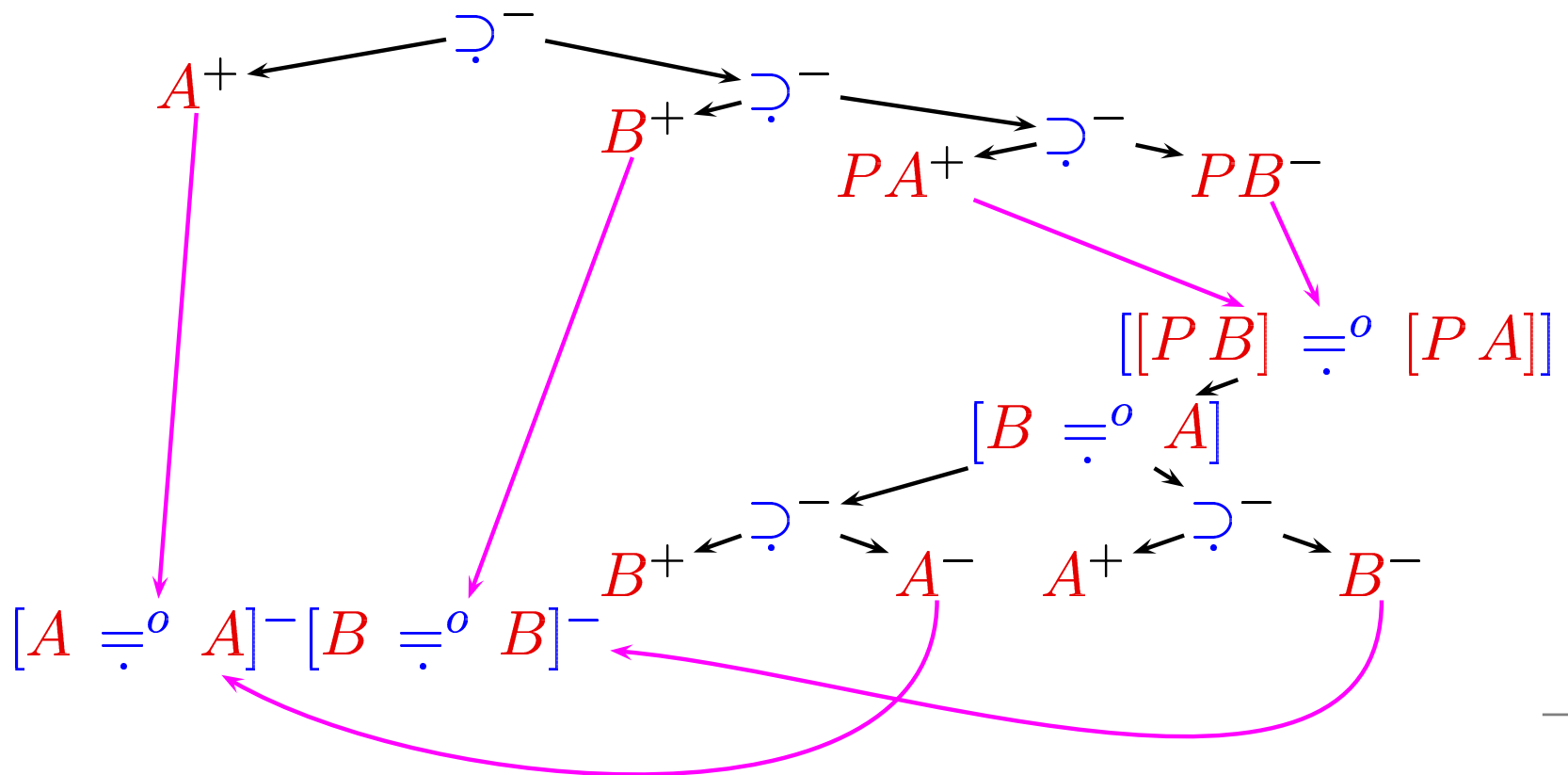
Extensional Expansion Dags

Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:



Extensional Expansion Dags

Reconsider $[A_o \supset \cdot B_o \supset \cdot [P_{oo} A] \supset [P B]]$:



Extensional Expansion Proofs

- Extensional Expansion Proofs =
(Closed) Extensional Expansion Dags with
No Vertical Paths

Extensional Expansion Proofs

- Extensional Expansion Proofs =
(Closed) Extensional Expansion Dags with
No Vertical Paths
- Extensional Expansion Proofs correspond to
Sequent Derivations.

Extensional Expansion Proofs

- Extensional Expansion Proofs =
(Closed) Extensional Expansion Dags with
No Vertical Paths
- Extensional Expansion Proofs correspond to
Sequent Derivations.
- Base Automated Search on
Extensional Expansion Dags

Extensional Search

- Basic Operations:

Extensional Search

- Basic Operations:
- Connections Between Atoms and Equations at Type ι

Extensional Search

- Basic Operations:
- Connections Between Atoms and Equations at Type ι
- Determine Expansion Variables via Unification

Extensional Search

- Basic Operations:
- Connections Between Atoms and Equations at Type ι
- Determine Expansion Variables via Unification
- Use Primsubs on Expansion Variables

Extensional Search

- Basic Operations:
- Connections Between Atoms and Equations at Type ι
- Determine Expansion Variables via Unification
- Use Primsubs on Expansion Variables
- (Optional) Set Constraints

Set Constraints

Reconsider $\exists s_{ol} \forall Z_l \cdot [s Z \equiv \cdot A Z \vee B Z]$

$$\left[\begin{array}{c} [s W^+] \\ A W^- \\ B W^- \end{array} \right] \vee \left[\begin{array}{c} A W^+ \\ s W^- \end{array} \right] \vee \left[\begin{array}{c} B W^+ \\ s W^- \end{array} \right]$$

Set Constraints

Reconsider $\exists s_{ol} \forall Z_l \cdot [s Z \equiv \cdot A Z \vee B Z]$

$$\left[\left[\begin{array}{c} s W^+ \\ A W^- \\ B W^- \end{array} \right] \vee \left[\begin{array}{c} A W^+ \\ s W^- \end{array} \right] \vee \left[\begin{array}{c} B W^+ \\ s W^- \end{array} \right] \right]$$

- View Each Vertical Path as a Set Constraint for s :

Set Constraints

Reconsider $\exists s_{ot} \forall Z_t \cdot [s Z \equiv \cdot A Z \vee B Z]$

$$\left[\left[\begin{array}{c} s W^+ \\ A W^- \\ B W^- \end{array} \right] \vee \left[\begin{array}{c} A W^+ \\ s W^- \end{array} \right] \vee \left[\begin{array}{c} B W^+ \\ s W^- \end{array} \right] \right]$$

- View Each Vertical Path as a Set Constraint for s :
- $\forall w_t \langle s w \rightarrow A w, B w \rangle$

Set Constraints

Reconsider $\exists s_{oi} \forall Z_i \cdot [s Z \equiv \cdot A Z \vee B Z]$

$$\left[\begin{array}{c} \left[\begin{array}{c} s W^+ \\ A W^- \\ B W^- \end{array} \right] \vee \left[\begin{array}{c} A W^+ \\ s W^- \end{array} \right] \vee \left[\begin{array}{c} B W^+ \\ s W^- \end{array} \right] \end{array} \right]$$

- View Each Vertical Path as a Set Constraint for s :
- $\forall w_i \langle s w \rightarrow A w, B w \rangle$
- $\forall w_i \langle A w \rightarrow s w \rangle$

Set Constraints

Reconsider $\exists s_{ol} \forall Z_l \cdot [s Z \equiv \cdot A Z \vee B Z]$

$$\left[\left[\begin{array}{c} s W^+ \\ A W^- \\ B W^- \end{array} \right] \vee \left[\begin{array}{c} A W^+ \\ s W^- \end{array} \right] \vee \left[\begin{array}{c} B W^+ \\ s W^- \end{array} \right] \right]$$

- View Each Vertical Path as a Set Constraint for s :
- $\forall w_l \langle s w \rightarrow A w, B w \rangle$
- $\forall w_l \langle A w \rightarrow s w \rangle$
- $\forall w_l \langle B w \rightarrow s w \rangle$

Set Constraints

Reconsider $\exists s_{ol} \forall Z_l \cdot [s Z \equiv \cdot A Z \vee B Z]$

$$\left[\begin{array}{c} [s W^+] \\ [A W^-] \\ [B W^-] \end{array} \vee \begin{array}{c} [A W^+] \\ [s W^-] \end{array} \vee \begin{array}{c} [B W^+] \\ [s W^-] \end{array} \right]$$

- Upper Bound Constraint:
- $\forall w_l \langle s w \rightarrow A w, B w \rangle$
- Maximal Solution: $A \cup B$

Set Constraints

Reconsider $\exists s_{ol} \forall Z_l \cdot [s Z \equiv \cdot A Z \vee B Z]$

$$\left[\begin{array}{c} \left[\begin{array}{c} s W^+ \\ A W^- \\ B W^- \end{array} \right] \vee \left[\begin{array}{c} A W^+ \\ s W^- \end{array} \right] \vee \left[\begin{array}{c} B W^+ \\ s W^- \end{array} \right] \end{array} \right]$$

- Lower Bound Constraints:
- $\forall w_l \langle A w \rightarrow s w \rangle$
- $\forall w_l \langle B w \rightarrow s w \rangle$
- Simultaneous Minimal Solution: $A \cup B$

Recursive Set Constraints

- Upper and Lower Bound Set Constraints may be Recursive.

Recursive Set Constraints

- Upper and Lower Bound Set Constraints may be Recursive.
- $\forall x, y \langle s_{oi}x, sy \rightarrow s[Fxy] \rangle$

Recursive Set Constraints

- Upper and Lower Bound Set Constraints may be Recursive.
- $\forall x, y \langle s_{oi}x, sy \rightarrow s[Fxy] \rangle$
- Least Solution: Empty Set (Trivial)

Recursive Set Constraints

- Upper and Lower Bound Set Constraints may be Recursive.
- $\forall x, y \langle s_{oi}x, sy \rightarrow s[Fxy] \rangle$
- Least Solution: Empty Set (Trivial)
- Combine with a Non-Recursive Lower Bound:

Recursive Set Constraints

- Upper and Lower Bound Set Constraints may be Recursive.
- $\forall x_l, y_l \langle s_{ol}x, s y \rightarrow s [F x y] \rangle$
- Least Solution: Empty Set (Trivial)
- Combine with a Non-Recursive Lower Bound:
- $\langle \cdot \rightarrow s A_l \rangle$

Recursive Set Constraints

- Upper and Lower Bound Set Constraints may be Recursive.
- $\forall x_l, y_l \langle s_{ol}x, s y \rightarrow s [F x y] \rangle$
- Least Solution: Empty Set (Trivial)
- Combine with a Non-Recursive Lower Bound:
- $\langle \cdot \rightarrow s A_l \rangle$
- Least Solution: Finite Binary Trees with Leaves A and Constructor F

Recursive Set Constraints

- Upper and Lower Bound Set Constraints may be Recursive.
- $\forall x_l, y_l \langle s_{ol}x, s y \rightarrow s [F x y] \rangle$
- Least Solution: Empty Set (Trivial)
- Combine with a Non-Recursive Lower Bound:
- $\langle \cdot \rightarrow s A_l \rangle$
- Least Solution: Finite Binary Trees with Leaves A and Constructor F
- TPS can Automatically Generate Such Solutions using Knaster-Tarski Fixed Point Theorem.

Embedded Occurrences

- Set Constraints May Contain Embedded Occurrences:

Embedded Occurrences

- Set Constraints May Contain Embedded Occurrences:
- $\forall x_i \langle L_{oi(oi)} \boxed{s} x \rightarrow s_{oi} x \rangle$

Embedded Occurrences

- Set Constraints May Contain Embedded Occurrences:
- $\forall x_i \langle L_{oi(oi)} s x \rightarrow s_{oi} x \rangle$
- Idea: Replace L with a “Monotone Approximation”
 $\mathbf{M} := [\lambda w \bigcup_{u \subseteq w} [L u]]$.

Embedded Occurrences

- Set Constraints May Contain Embedded Occurrences:
- $\forall x_l \langle L_{ol(ol)} s x \rightarrow s_{ol} x \rangle$
- Idea: Replace L with a “Monotone Approximation”
 $\mathbf{M} := [\lambda w \bigcup_{u \subseteq w} [L u]]$.
- \mathbf{M} is Monotone for Syntactic Reasons.

Embedded Occurrences

- Set Constraints May Contain Embedded Occurrences:
- $\forall x_i \langle L_{oi(oi)} s x \rightarrow s_{oi} x \rangle$
- Idea: Replace L with a “Monotone Approximation”
 $\mathbf{M} := [\lambda w \bigcup_{u \subseteq w} [L u]]$.
- \mathbf{M} is Monotone for Syntactic Reasons.
- $[L w] \subseteq [\mathbf{M} w]$

Embedded Occurrences

- Set Constraints May Contain Embedded Occurrences:
- $\forall x_i \langle L_{oi(oi)} s x \rightarrow s_{oi} x \rangle$
- Idea: Replace L with a “Monotone Approximation”
 $\mathbf{M} := [\lambda w \bigcup_{u \subseteq w} [L u]]$.
- \mathbf{M} is Monotone for Syntactic Reasons.
- $[L w] \subseteq [\mathbf{M} w]$
- Use Knaster-Tarski to find Least W with $[\mathbf{M} W] \subseteq W$.

Embedded Occurrences

- Set Constraints May Contain Embedded Occurrences:
- $\forall x_i \langle L_{oi(oi)} s x \rightarrow s_{oi} x \rangle$
- Idea: Replace L with a “Monotone Approximation”
 $\mathbf{M} := [\lambda w \bigcup_{u \subseteq w} [L u]]$.
- \mathbf{M} is Monotone for Syntactic Reasons.
- $[L w] \subseteq [\mathbf{M} w]$
- Use Knaster-Tarski to find Least W with $[\mathbf{M} W] \subseteq W$.
- $[L W] \subseteq W$.

Knaster-Tarski Theorem

THM2F $\forall K_{ol(ol)} \cdot \forall x_{ol} \forall y_{ol} [x \subseteq y \supset K x \subseteq K y]$
 $\supset \exists u_{ol} \cdot K u =^{ol} u$

If K is monotone, then K has a fixed point.

Knaster-Tarski Theorem

THM2F $\forall K_{\alpha} \cdot \forall x \forall y [x \subseteq y \supset Kx \subseteq Ky]$
 $\supset \exists u \cdot Ku =^{\alpha} u$

- Recursive Set Constraint for w_{α} :

$$\forall x \langle K_{\alpha} w x \rightarrow w x \rangle$$

Knaster-Tarski Theorem

THM2F $\forall K_{\omega_l(\omega_l)} \cdot \forall x_{\omega_l} \forall y_{\omega_l} [x \subseteq y \supset K x \subseteq K y]$
 $\supset \exists u_{\omega_l} \cdot K u =^{\omega_l} u$

- Recursive Set Constraint for w_{ω_l} :

$$\forall x_{\omega_l} \langle K_{\omega_l(\omega_l)} w x \rightarrow w x \rangle$$

- The Natural Monotone Set Function to Consider is K .

Knaster-Tarski Theorem

THM2F $\forall K_{ol(ol)} \cdot \forall x_{ol} \forall y_{ol} [x \subseteq y \supset K x \subseteq K y]$
 $\supset \exists u_{ol} \cdot K u =^{ol} u$

- Recursive Set Constraint for w_{ol} :
 $\forall x_{ol} \langle K_{ol(ol)} w x \rightarrow w x \rangle$
- The Natural Monotone Set Function to Consider is K .
- But We Cannot Syntactically Determine If K is Monotone.

Knaster-Tarski Theorem

THM2F $\forall K_{ol(ol)} \cdot \forall x_{ol} \forall y_{ol} [x \subseteq y \supset K x \subseteq K y]$
 $\supset \exists u_{ol} \cdot K u =^{ol} u$

- Recursive Set Constraint for w_{ol} :
 $\forall x_{ol} \langle K_{ol(ol)} w x \rightarrow w x \rangle$
- The Natural Monotone Set Function to Consider is K .
- But We Cannot Syntactically Determine If K is Monotone.
- Use the “Monotone Approximation” M

Knaster-Tarski Theorem

THM2F $\forall K_{ol(ol)} \cdot \forall x_{ol} \forall y_{ol} [x \subseteq y \supset K x \subseteq K y]$
 $\supset \exists u_{ol} \cdot K u =^{ol} u$

- Recursive Set Constraint for w_{ol} :
 $\forall x_l \langle K_{ol(ol)} w x \rightarrow w x \rangle$
- The Natural Monotone Set Function to Consider is K .
- But We Cannot Syntactically Determine If K is Monotone.
- Use the “Monotone Approximation” M
- TPS Solves for W with $MW \subseteq W$.

Knaster-Tarski Theorem

THM2F $\forall K_{ol(ol)} \cdot \forall x_{ol} \forall y_{ol} [x \subseteq y \supset K x \subseteq K y]$
 $\supset \exists u_{ol} \cdot K u =^{ol} u$

- Recursive Set Constraint for w_{ol} :
 $\forall x_l \langle K_{ol(ol)} w x \rightarrow w x \rangle$
- The Natural Monotone Set Function to Consider is K .
- But We Cannot Syntactically Determine If K is Monotone.
- Use the “Monotone Approximation” M
- TPS Solves for W with $MW \subseteq W$.
- TPS Proves $KW = W$ Using Monotonicity of K (Hypothesis).

Conclusion

- Extensionality Permits Restrictions on Primsubs

Conclusion

- Extensionality Permits Restrictions on Primsubs
- Cut-Free Sequent Calculus for each \mathcal{S} Fragment of Extensional Type Theory with Equality Reasoning.

Conclusion

- Extensionality Permits Restrictions on Primsubs
- Cut-Free Sequent Calculus for each \mathcal{S} Fragment of Extensional Type Theory with Equality Reasoning.
- Every Theorem of Extensional Type Theory has an Extensional Expansion Proof Based on an Extensional Expansion Dag

Conclusion

- Extensionality Permits Restrictions on Primsubs
- Cut-Free Sequent Calculus for each \mathcal{S} Fragment of Extensional Type Theory with Equality Reasoning.
- Every Theorem of Extensional Type Theory has an Extensional Expansion Proof Based on an Extensional Expansion Dag
- Automated Search using Extensional Expansion Dags

Conclusion

- Extensionality Permits Restrictions on Primsubs
- Cut-Free Sequent Calculus for each \mathcal{S} Fragment of Extensional Type Theory with Equality Reasoning.
- Every Theorem of Extensional Type Theory has an Extensional Expansion Proof Based on an Extensional Expansion Dag
- Automated Search using Extensional Expansion Dags
- Using Set Constraints Also Helps Instantiate Set Variables

Conclusion

- Extensionality Permits Restrictions on Primsubs
- Cut-Free Sequent Calculus for each \mathcal{S} Fragment of Extensional Type Theory with Equality Reasoning.
- Every Theorem of Extensional Type Theory has an Extensional Expansion Proof Based on an Extensional Expansion Dag
- Automated Search using Extensional Expansion Dags
- Using Set Constraints Also Helps Instantiate Set Variables
- Search Procedures Based on these Ideas are Implemented in TPS