

Parsing Correctness Criterion for Second order Multiplicative Linear Logic

Adrien Ragué^{1 2}

Thomas Seiller¹

Lorenzo Tortora de Falco²

¹ Université Sorbonne Paris Nord
(LIPN)

² Università Degli Studi Roma Tre
(Dip. Matematica e Fisica)

PLAN

I MLL2 Nets : challenges

II Typed vs untyped nets

- OUR CRITERION

III Nets : desequentializing proofs

- OUR CRITERION

IV PARSING strategy

V PARSING (naive)

VI PARSING with schemes

PLAN

I MLL2 Nets : challenges



II Typed vs untyped nets

- OUR CRITERION

III Nets : desequentializing proofs

- OUR CRITERION

IV PARSING strategy

V PARSING (naive)

VI PARSING with schemes

I MLL2 Nets : challenges

- 1- MLL2 Sequent calculus
- 2- MLL2 typed nets
- 3- Quantifiers & connectives
- 4- Quantifiers & types
- 5 - Dimensions
- 6 - MLL2⁺ Sequent calculus

I MLL2 Nets : challenges

1- MLL2 Sequent calculus

2- MLL2 typed nets

3- Quantifiers & connectives

4- Quantifiers & types

5- Dimensions

6 - MLL2⁺ Sequent calculus

MU2 Sequent Calculus

$\frac{}{A_1 A^x}$

MU2 Sequent Calculus

$$\frac{}{A, A^{\perp}} ax$$

$$\frac{\Gamma, A \quad \Delta, A^{\perp}}{\Gamma, \Delta} cut$$

MU2 Sequent Calculus

$$\frac{}{A, A^\perp} ax$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} cut$$

MU2 Sequent Calculus

$$\frac{}{A, A^{\perp}} ax$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, A^{\perp}}{\Gamma, \Delta} cut$$

$$\frac{\Gamma, A[x/y]}{\Gamma, \forall z A[z/y]} \forall$$

$x \notin free(\Gamma)$

$$\frac{\Gamma, A[B/y]}{\Gamma, \exists z A[z/y]} \exists$$

I MLL2 Nets : challenges

1- MLL2 Sequent calculus

2- MLL2 typed nets

3- Quantifiers & connectives

4- Quantifiers & types

5- Dimensions

6 - MLL2⁺ Sequent calculus

ML2 Typed Nets

MILL2 Typed Nets

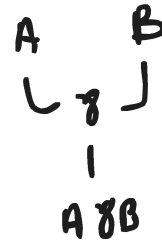
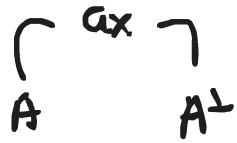
$$\begin{array}{c} \lrcorner \quad \text{ax} \quad \lrcorner \\ A \quad \quad A^\perp \end{array}$$

MIL2 Typed Nets

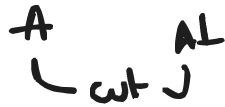
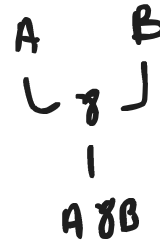
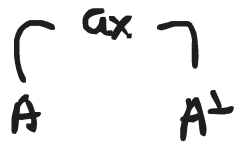
$$\begin{array}{c} \text{ } \quad \text{ax} \quad \text{ } \\ \text{ } \quad \text{ } \quad \text{ } \\ \text{A} \quad \quad \text{A}^{\perp} \end{array}$$

$$\begin{array}{cc} \text{A} & \text{B} \\ \text{ } \quad \text{ } & \text{ } \\ \text{ } \quad \text{ } & \text{ } \\ \text{ } \quad \text{ } & \text{ } \\ \text{A} \otimes \text{B} \end{array}$$

MIL2 Typed Nets



MIL2 Typed Nets



ML2 Typed Nets

$$\begin{array}{c} \text{Ax} \\ \text{A} \quad \text{A}^{\perp} \end{array}$$

A B
L O J
I
A O B

$$\begin{array}{cc} A & B \\ \swarrow & \searrow \\ & \delta \\ & | \\ & A \delta B \end{array}$$

A AL
└── cut ─┘

$$\begin{array}{c} A[x/y] \\ | \\ \forall \\ | \\ \forall z A[z/y] \end{array}$$

MIL2 Typed Nets

$$\left(\begin{array}{c} \text{Ax} \\ A \end{array} \right) \quad \left(\begin{array}{c} \\ A^{\perp} \end{array} \right)$$

$$\begin{array}{cc} A & B \\ \downarrow \otimes \downarrow & \\ A \otimes B \end{array}$$

$$\begin{array}{cc} A & B \\ \downarrow \otimes \downarrow & \\ A \otimes B \end{array}$$

$$\begin{array}{cc} A & A^{\perp} \\ \downarrow \text{cut} \downarrow & \end{array}$$

$$\begin{array}{c} A[x/y] \\ \downarrow \\ \forall \\ \downarrow \\ \forall z A[z/y] \end{array}$$

$$\begin{array}{c} A[B/y] \\ \downarrow \\ \exists \\ \downarrow \\ \exists z A[z/y] \end{array}$$

I MLL2 Nets : challenges

1- MLL2 Sequent calculus

2- MLL2 typed nets

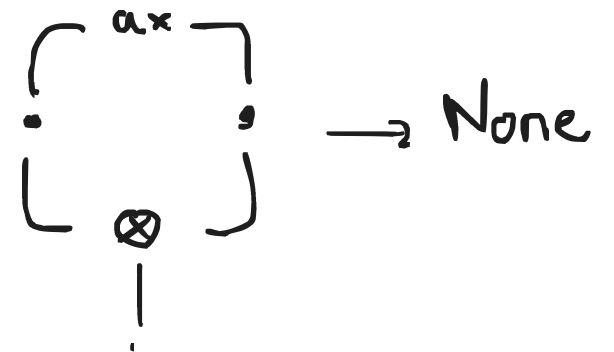
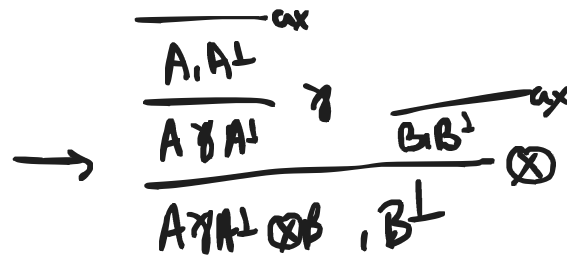
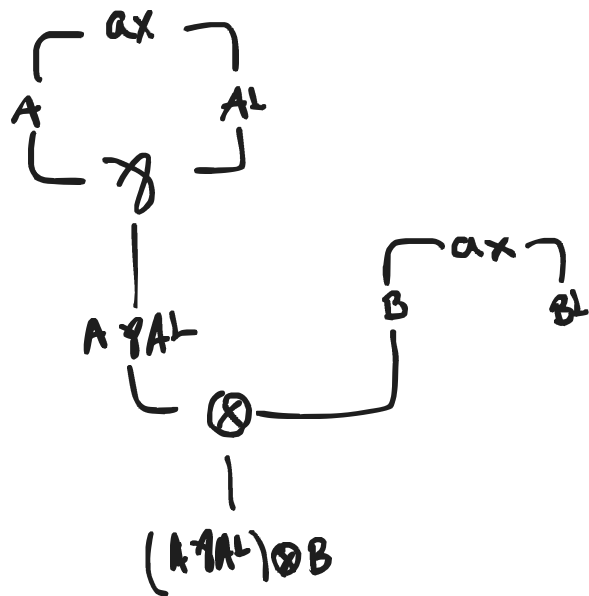
3- Quantifiers & connectives

4- Quantifiers & types

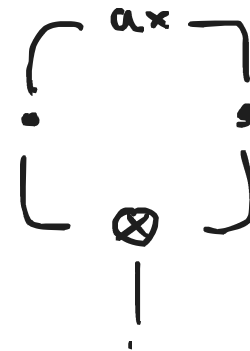
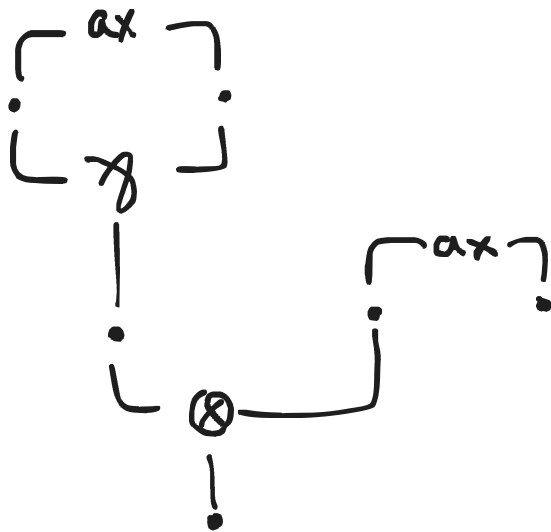
5- Dimensions

6 - MLL2⁺ Sequent calculus

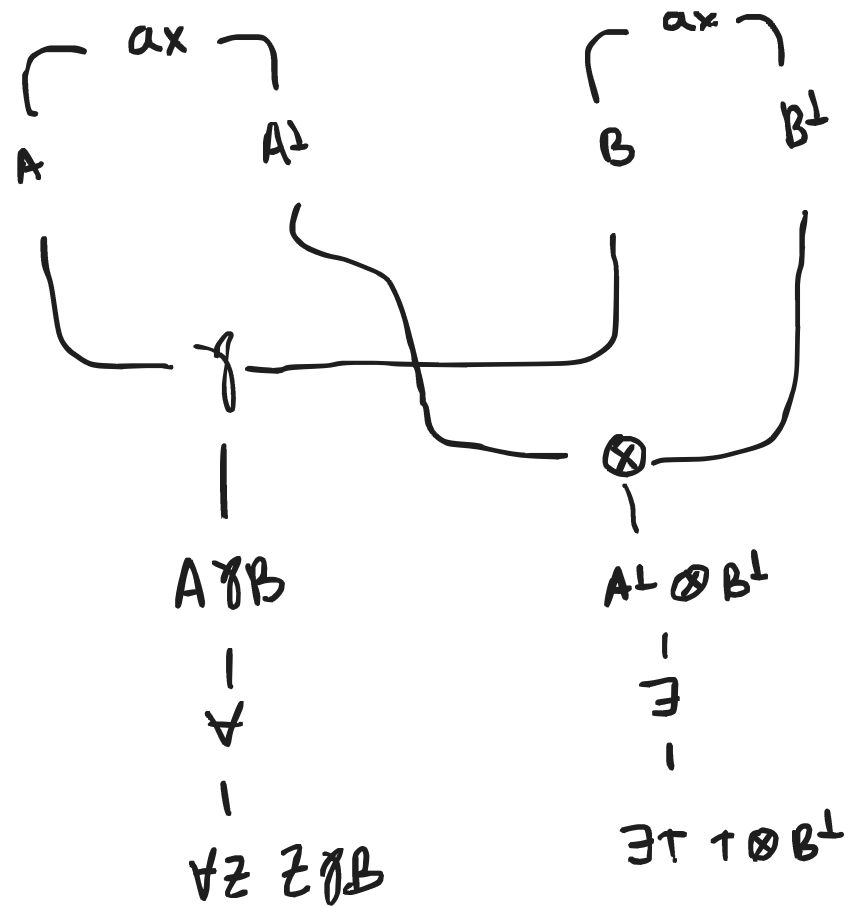
QUANTIFIERS AND CONNECTIVE



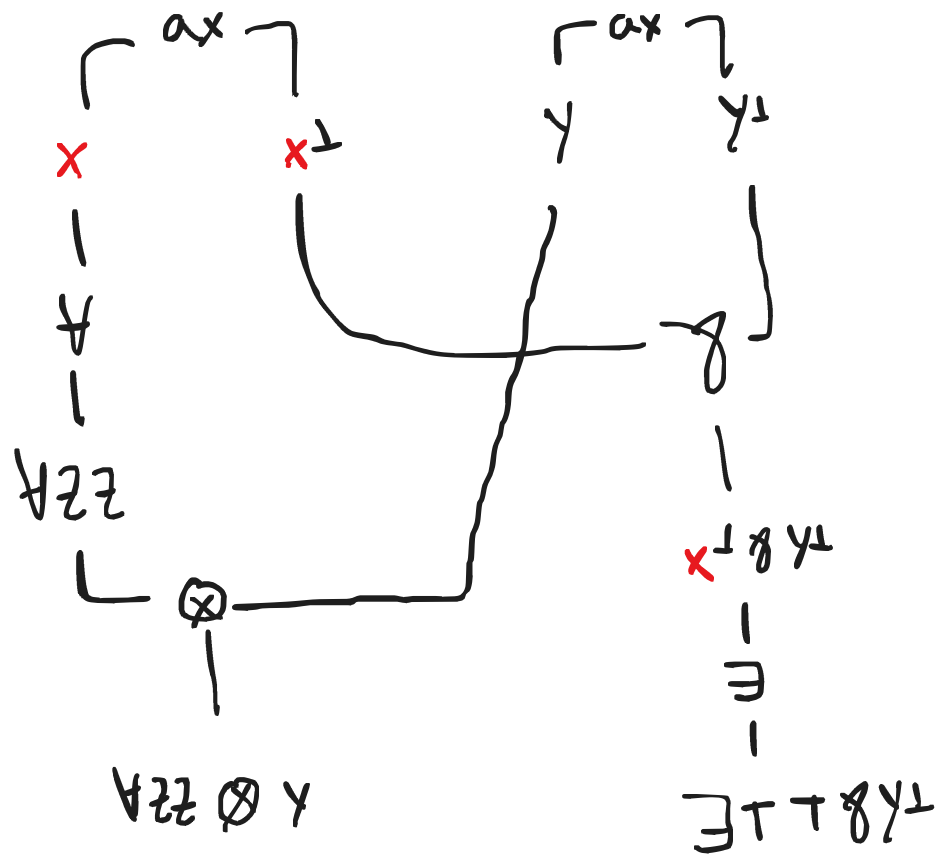
QUANTIFIERS AND CONNECTIVE



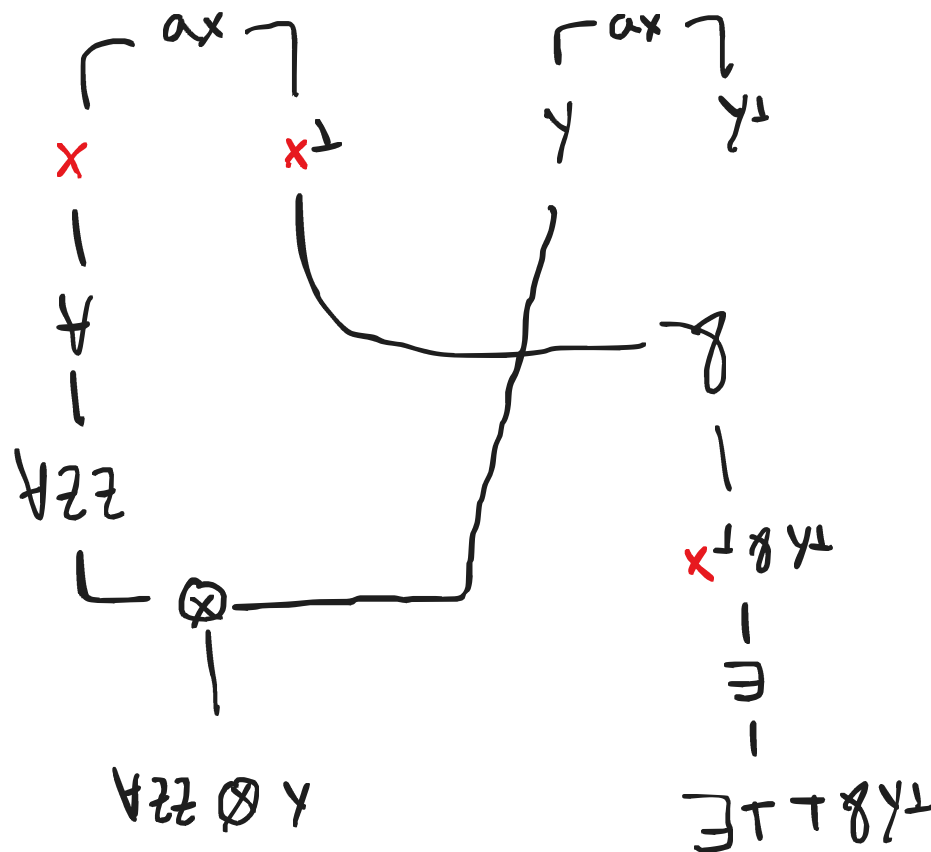
QUANTIFIERS AND CONNECTIVE



QUANTIFIERS AND CONNECTIVE

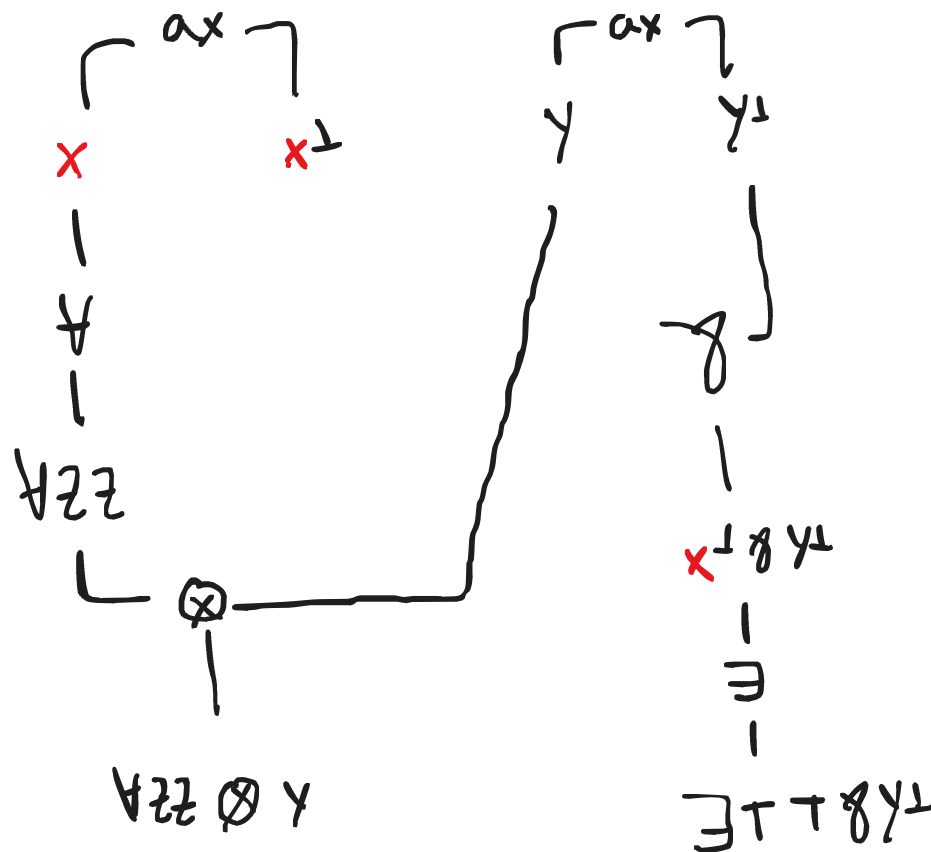


QUANTIFIERS AND CONNECTIVE



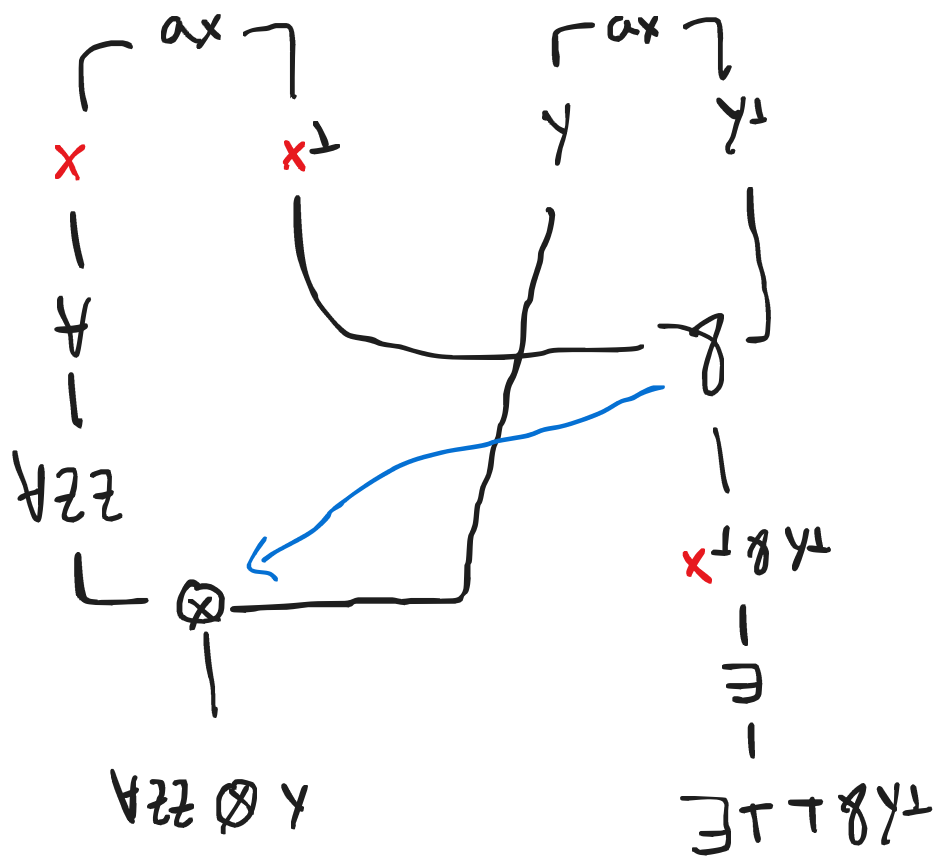
is ACC

QUANTIFIERS AND CONNECTIVE

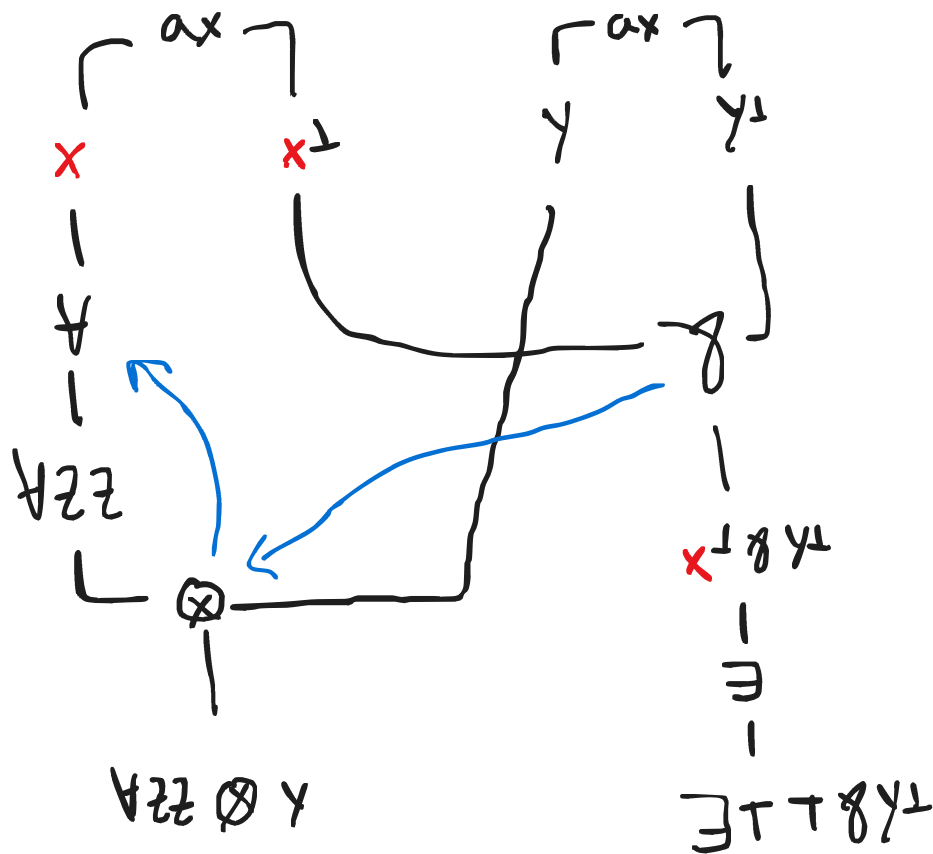


is ACC

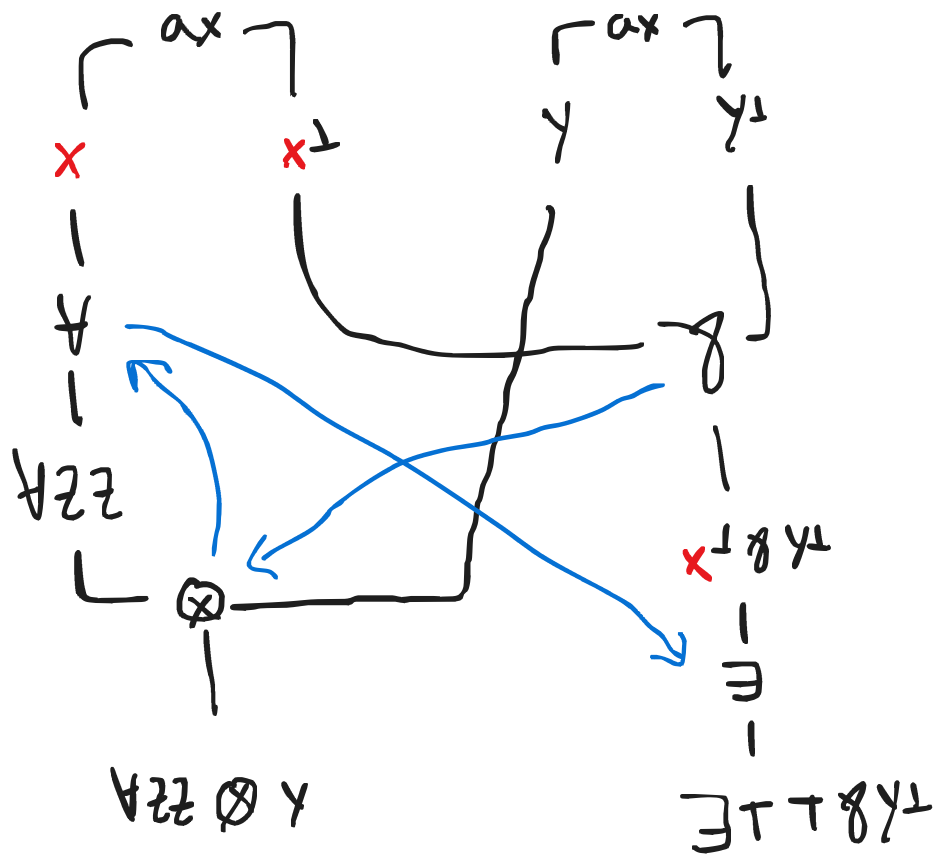
QUANTIFIERS AND CONNECTIVE



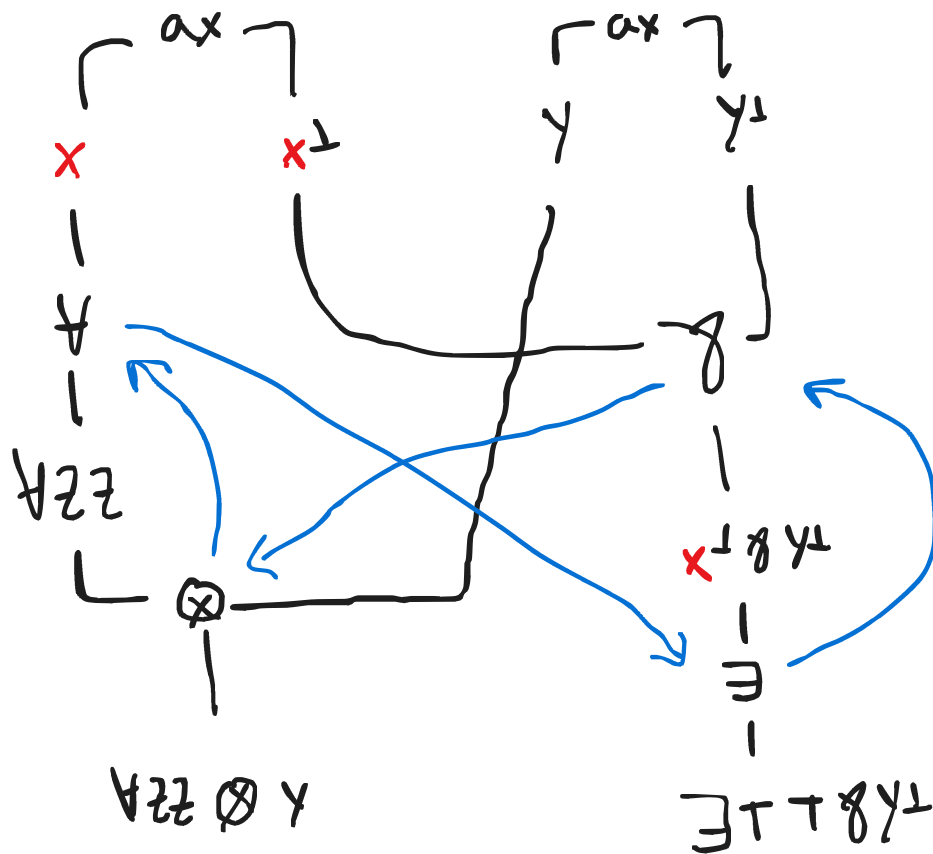
QUANTIFIERS AND CONNECTIVE



QUANTIFIERS AND CONNECTIVE



QUANTIFIERS AND CONNECTIVE



I MLL2 Nets : challenges

1- MLL2 Sequent calculus

2- MLL2 typed nets

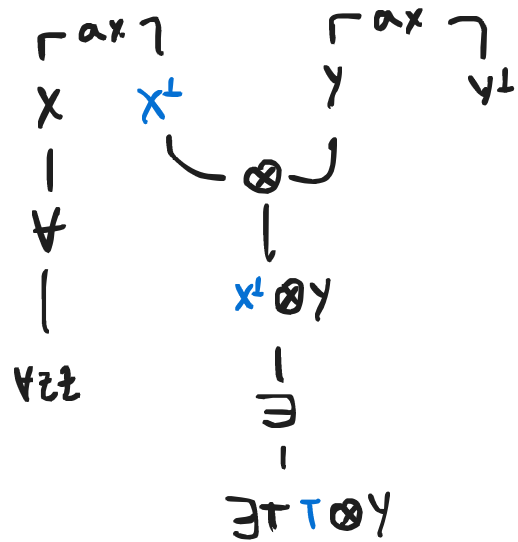
3- Quantifiers & connectives

4- Quantifiers & types

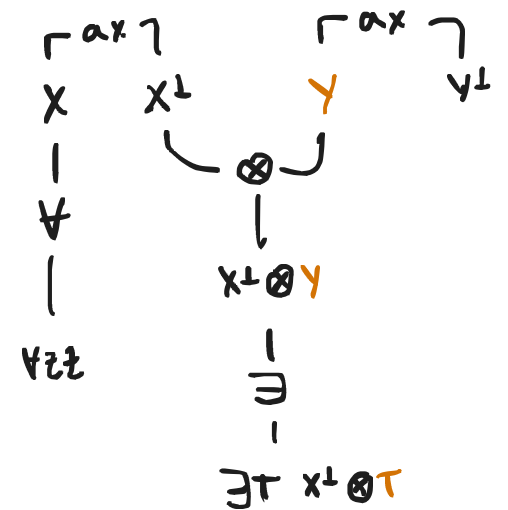
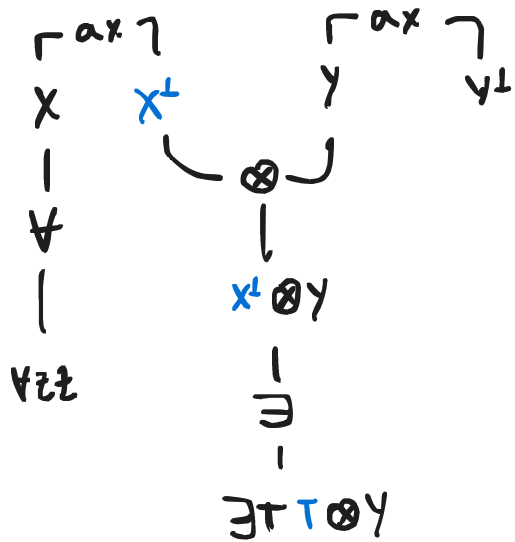
5 - Dimensions

6 - MLL2⁺ Sequent calculus

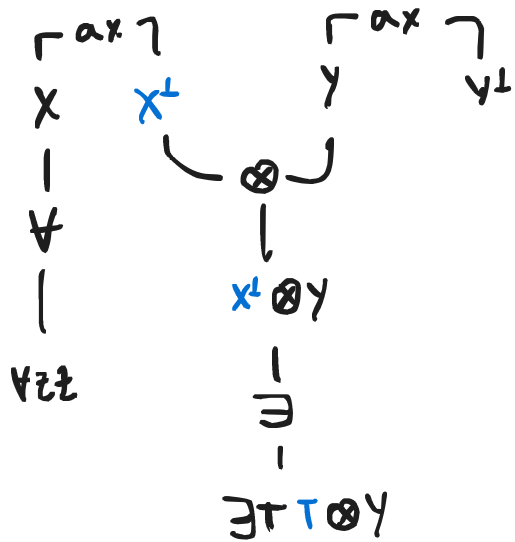
TYPES & QUANTIFIERS



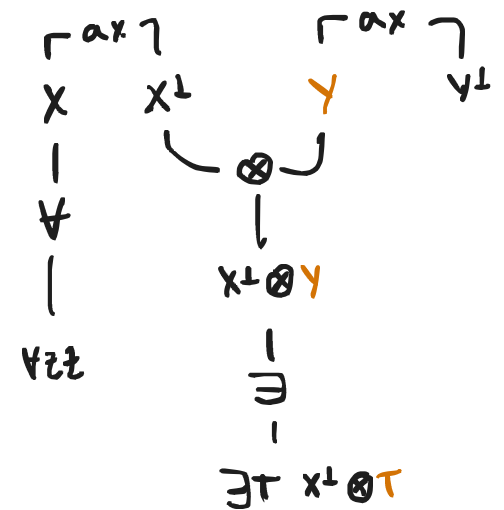
TYPES & QUANTIFIERS



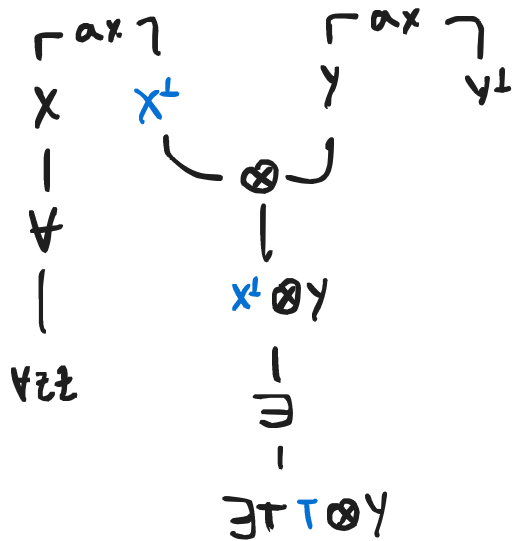
TYPES & QUANTIFIERS



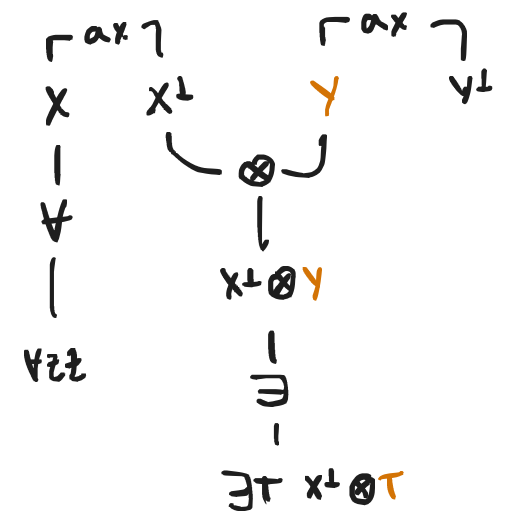
correct



TYPES & QUANTIFIERS

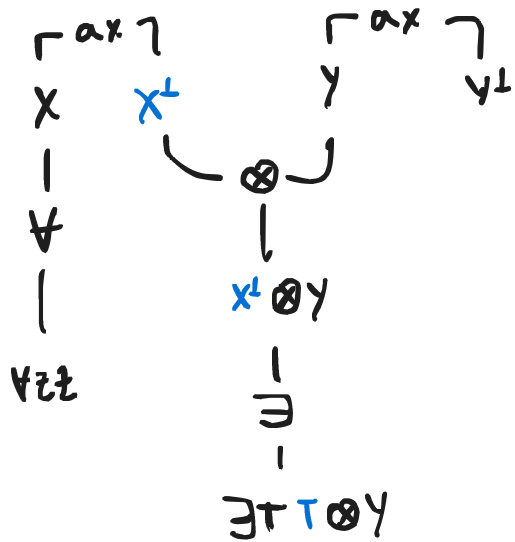


correct

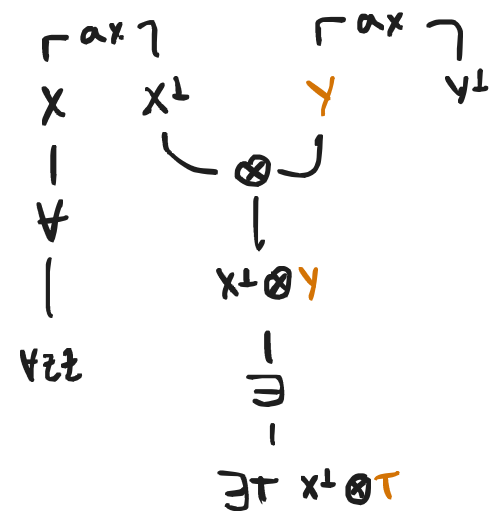
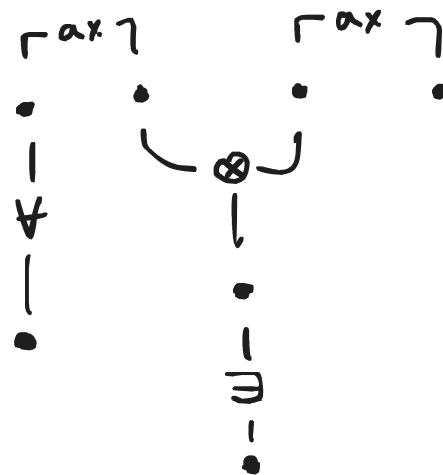


incorrect

TYPES & QUANTIFIERS

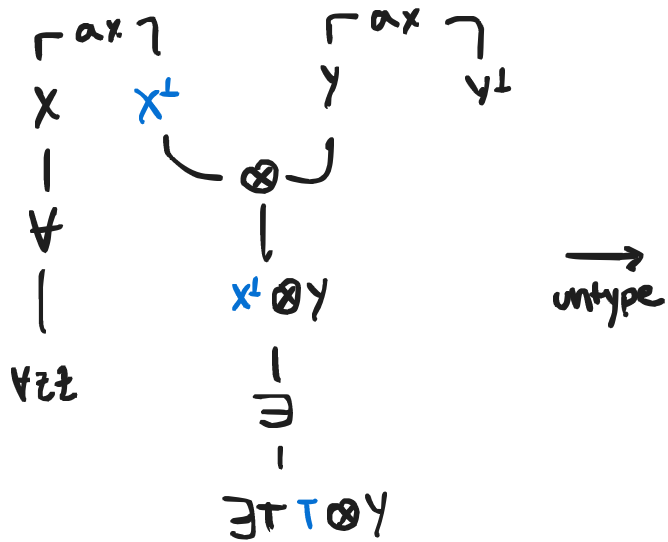


correct

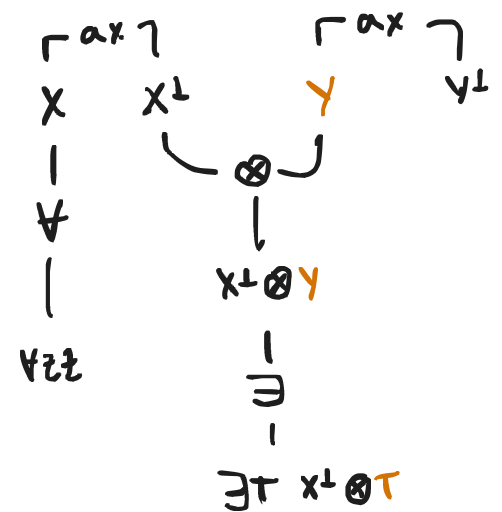
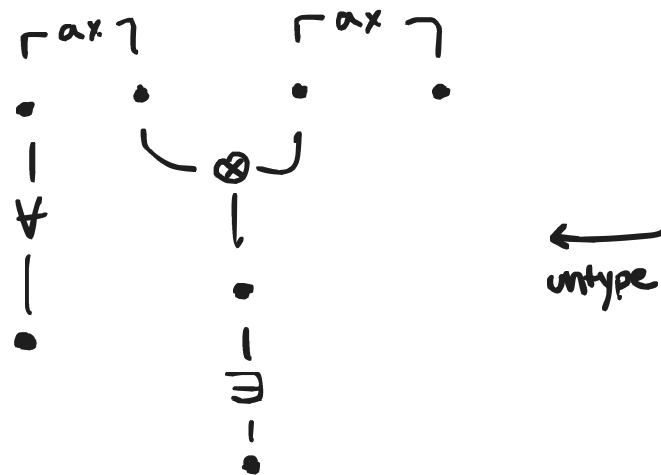


incorrect

TYPES & QUANTIFIERS

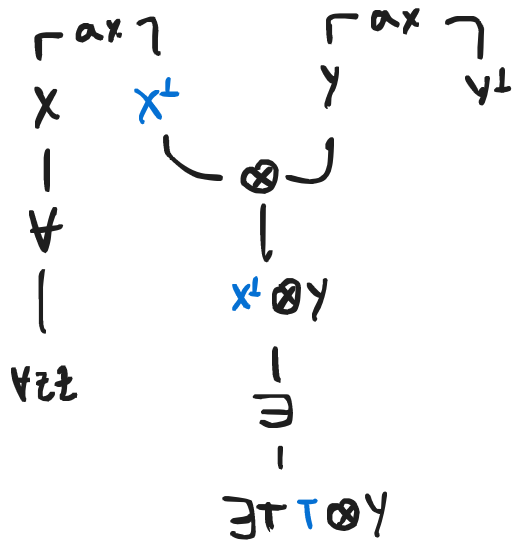


correct

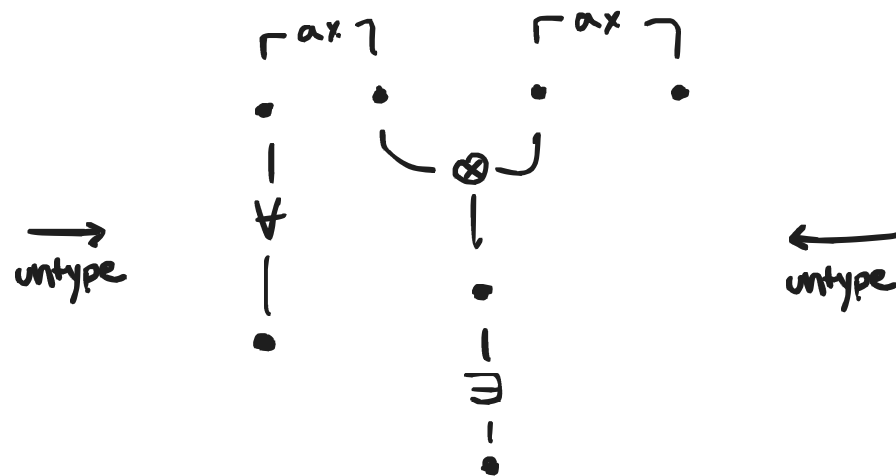


incorrect

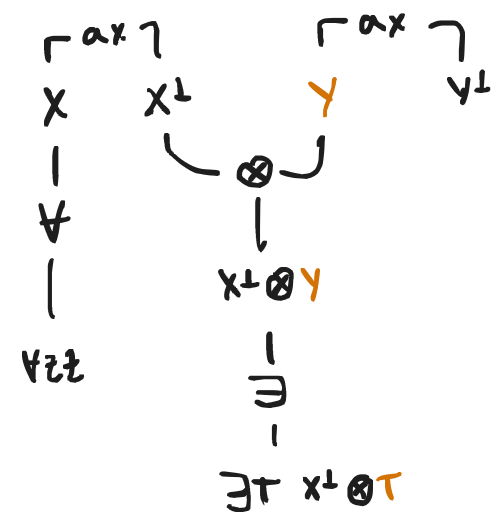
TYPES & QUANTIFIERS



correct



Need additional information $\rightarrow$ POINTERS



incorrect

I MLL2 Nets : challenges

1- MLL2 Sequent calculus

2- MLL2 typed nets

3- Quantifiers & connectives

4- Quantifiers & types

5 - Daimons

6 - MLL2⁺ Sequent calculus

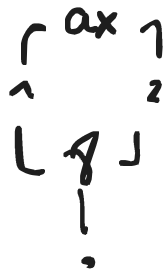
DAIMONS

DAIMONS

$\mathbb{H}$: generalised axioms

DAIMONS

$\mathbb{H}$: generalised axioms



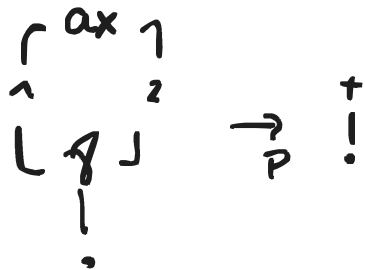
DAIMONS

$\mathbb{H}$: generalised axioms

$$\begin{array}{c} \text{ax} \\ \text{1} \quad \text{2} \\ \text{L} \quad \text{J} \\ \text{!} \end{array} \rightarrow \begin{array}{c} \text{+} \\ \text{P} \\ \text{!} \end{array}$$

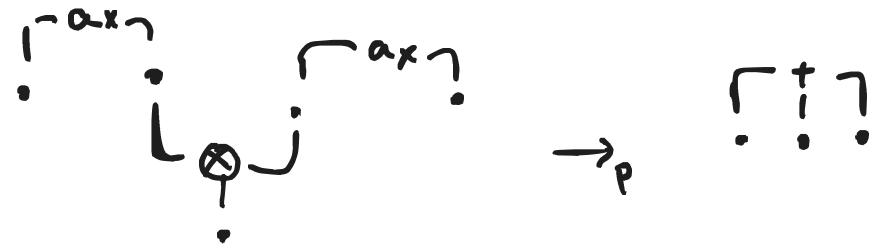
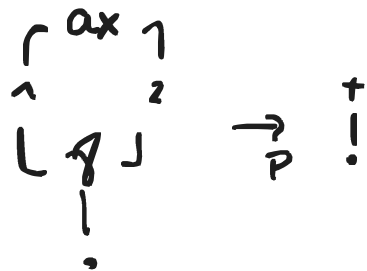
DAIMONS

$\mathbb{H}$: generalised axioms



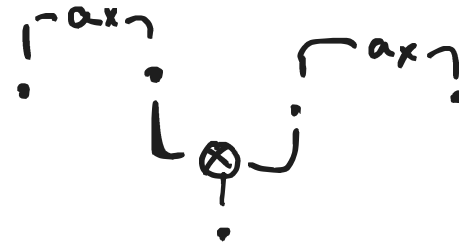
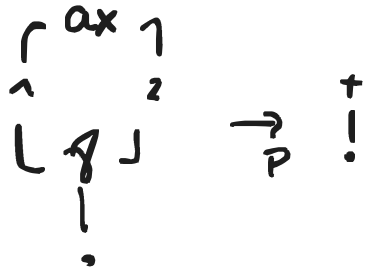
DAIMONS

$\mathbb{H}$: generalised axioms



DAIMONS

$\vdash$: generalised axioms = "partial proof"



I MLL2 Nets : challenges

- 1- MLL2 Sequent calculus
- 2- MLL2 typed nets
- 3- Quantifiers & connectives
- 4- Quantifiers & types
- 5- Dimensions
- 6 - MLL2⁺ Sequent calculus

MLL2⁺ Sequent Calculus

MLL2[†] Sequent Calculus

$$\frac{}{A, A^\perp} ax$$

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{}{A, A^\perp} \text{ax}$$

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{}{A, A^\perp} \text{ax}$$

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{}{A, A^\perp} \text{ax}$$

MLL2^{*} Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{}{A, A^\perp} \text{ax}$$

$$\frac{\Gamma, A[B/x]}{\Gamma, \exists z A[z/x]} \exists$$

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{}{A, A^\perp} \text{ax}$$

$$\frac{\Gamma, A[B/x]}{\Gamma, \exists z A[z/x]} \exists$$

$\uparrow$
 fresh

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{}{A, A^\perp} \text{ax}$$

$$\frac{\Gamma, A[B/x]}{\Gamma, \exists z A[z/x]} \exists$$

↑
fresh

$$\frac{\Gamma, A[y/x]}{\Gamma, \forall z A[z/x]} \forall$$

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{}{A, A^\perp} \text{ax}$$

$$\frac{\Gamma, A[B/x]}{\Gamma, \exists z A[z/x]} \exists$$

↑
fresh

$$\frac{\Gamma, A[y/x]}{\Gamma, \forall z A[z/x]} \forall$$

↑
fresh

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{}{A, A^\perp} \text{ax}$$

$$\frac{\Gamma, A[B/x]}{\Gamma, \exists z A[z/x]} \exists$$

↑
fresh

$$\frac{\Gamma, A[y/x]}{\Gamma, \forall z A[z/x]} \forall$$

↑
fresh

↑
y ∉ FV(Γ)

MLL2^{*} Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

$$\frac{}{A, A^\perp} \text{ax}$$

$$\frac{\Gamma, A[B/x]}{\Gamma, \exists z A[z/x]} \exists$$

↑
fresh

$$\frac{\Gamma, A[y/x]}{\Gamma, \forall z A[z/x]} \forall$$

↑
fresh

↑
y ∉ FV(Γ)

proof = tree defined by induction

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

~~$$\frac{}{A, A^\perp} \text{ax}$$~~

$$\frac{\Gamma, A[B/x]}{\Gamma, \exists z A[z/x]} \exists$$

↑ fresh

$$\frac{\Gamma, A[y/x]}{\Gamma, \forall z A[z/x]} \forall$$

↑ fresh

y ∉ FV(Γ)

proof = tree defined by induction

MLL2[†] Sequent Calculus

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut}$$

$$\frac{\Gamma, A, B}{\Gamma, A \wp B} \wp$$

$$\frac{\Gamma, A \quad \Delta, B}{\Gamma, A \otimes B, \Delta} \otimes$$

~~$$\frac{}{A, A^\perp} \text{ax}$$~~

$$\frac{}{\Gamma} \dagger \quad (\text{any sequent})$$

$$\frac{\Gamma, A[B/x]}{\Gamma, \exists z A[z/x]} \exists$$

↑ fresh

$$\frac{\Gamma, A[y/x]}{\Gamma, \forall z A[z/x]} \forall$$

↑ fresh y ∉ FV(Γ)

proof = tree defined by induction

PLAN

I MLL2 Nets : challenges

II Typed vs untyped nets 

- OUR CRITERION

III Nets : desequentializing proofs

- OUR CRITERION

IV PARSING strategy

V PARSING (naive)

VI PARSING with schemes

TYPED VS UNTYPED

- 1 - Untyped nets (naive)
- 2 - How to type a net
- 3 - is atomic typing enough?
- 4 - typing is non-trivial
- 5 - Comparative
- 6 - Parsing as type inference
- 7 - More incorrectness

TYPED VS UNTYPED

1 - Untyped nets (naive)

2 - How to type a net

3 - is atomic typing enough?

4 - typing is non-trivial

5 - Comparative

6 - Parsing as type inference

7 - More incorrectness

MLL2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, e)



position $\uparrow$ V

links $\uparrow$ E

ML2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)

$s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\nearrow$

links $\uparrow$

ML2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)

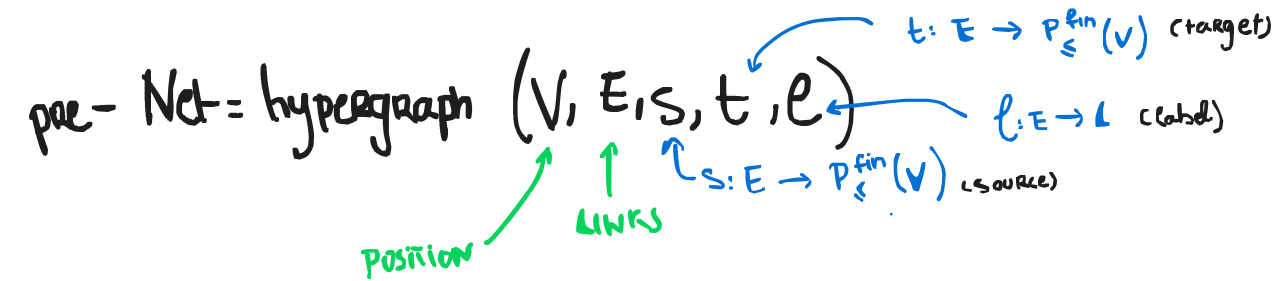
$\ell: E \rightarrow L$ (label)

$s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$

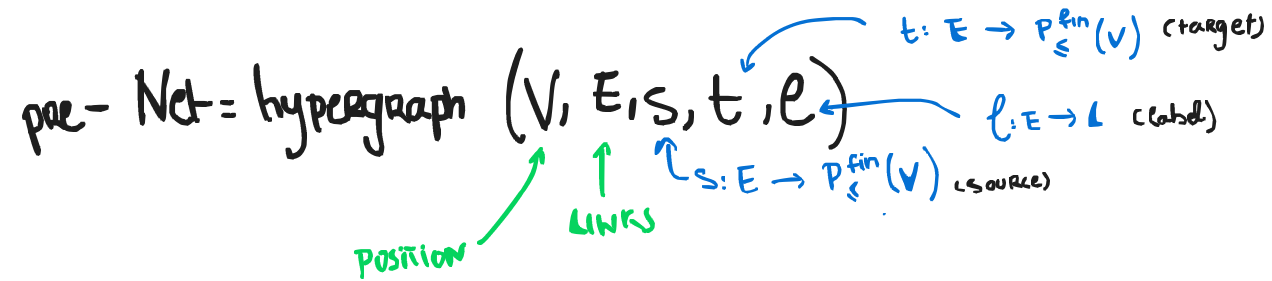
links $\uparrow$

MLL2+ Prae-nets



Links:

ML2+ Prae-nets



Links:

$$\begin{matrix} 1 & \dots & n \\ \uparrow & & \uparrow \\ 1 & \dots & n \end{matrix}$$

ML2+ Prae-nets

pra-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\begin{matrix} 1 & \dots & n \\ \uparrow & & \uparrow \\ 1 & \dots & n \end{matrix}$

$\begin{matrix} p_1 & p_2 \\ \downarrow & \downarrow \\ \downarrow & \downarrow \\ \downarrow & \downarrow \\ p \end{matrix}$

ML2+ Prae-nets

pra-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\{ \overset{+}{1} \dots n \}$

p_1 p_2
 $\{ \} \cup$
 $\downarrow$
 p

p_1 p_2
 $\{ \otimes \} \cup$
 $\downarrow$
 p

ML2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\begin{matrix} \lceil & + & \rceil \\ 1 & \dots & n \end{matrix}$

$\begin{matrix} p_1 & & p_2 \\ \downarrow & & \downarrow \\ \downarrow & \otimes & \downarrow \\ & p & \end{matrix}$

$\begin{matrix} p_1 & & p_2 \\ \downarrow & & \downarrow \\ \downarrow & \otimes & \downarrow \\ & p & \end{matrix}$

$\begin{matrix} p \\ \downarrow \\ \downarrow \\ \downarrow \\ q \end{matrix}$

MLL2+ Prae-nets

pra-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\leq}^{fin}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{fin}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\begin{matrix} \lceil & + & \rceil \\ 1 & \dots & n \end{matrix}$

$\begin{matrix} p_1 & & p_2 \\ \lceil & & \rceil \\ & \otimes & \\ & | & \\ & p & \end{matrix}$

$\begin{matrix} p_1 & & p_2 \\ \lceil & & \rceil \\ & \otimes & \\ & | & \\ & p & \end{matrix}$

$\begin{matrix} p \\ | \\ \vee \\ | \\ q \end{matrix}$

$\begin{matrix} p \\ | \\ \exists \\ | \\ q \end{matrix}$

MLL2+ Prae-nets

pra-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\{ \overset{+}{1} \dots n \}$

p_1 p_2
 $\{ \dots \}$
 p

p_1 p_2
 $\{ \otimes \}$
 p

p
 $\{ \forall \}$
 q

p
 $\{ \exists \}$
 q

p_1 p_2
 $\{ \text{cut} \}$

ML2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\{ \overset{+}{1} \dots n \}$

p_1 p_2
 $\{ \dots \}$
 p

p_1 p_2
 $\{ \otimes \}$
 p

p
 $\{ \forall \}$
 q

p
 $\{ \exists \}$
 q

p_1 p_2
 $\{ \text{cut} \}$

examples

ML2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\uparrow^+ \downarrow$
1 ... n

p_1 p_2
 $\downarrow$ $\downarrow$
 $\downarrow$
 p

p_1 p_2
 $\downarrow$ $\downarrow$
 $\downarrow$
 p

p
 $\downarrow$
 $\downarrow$
 q

p
 $\downarrow$
 $\downarrow$
 q

p_1 p_2
 $\downarrow$ cut $\downarrow$

examples

$\uparrow^+ \downarrow$
 p_1 p_2 p_3
 $\downarrow$ $\downarrow$ $\downarrow$
 $\downarrow$
 q

MLL2+ Pae-nets

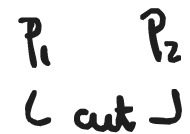
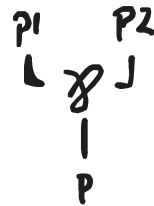
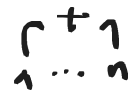
pre-Net = hypergraph (V, E, s, t, ℓ)

- $t: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (target)
- $\ell: E \rightarrow L$ (label)
- $s: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (source)

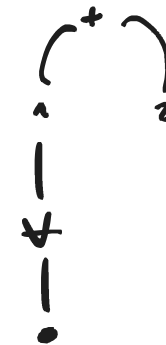
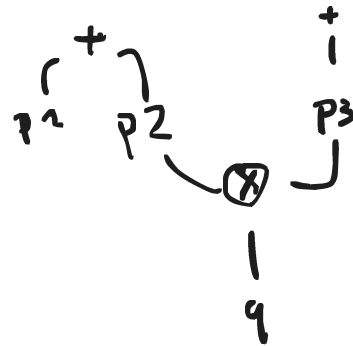
position (green arrow pointing to V)

links (green arrow pointing to E)

Links :



examples



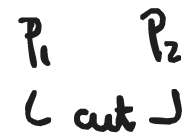
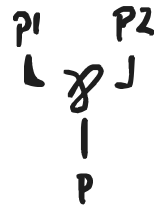
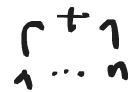
ML2+ Prae-nets

$\text{prae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

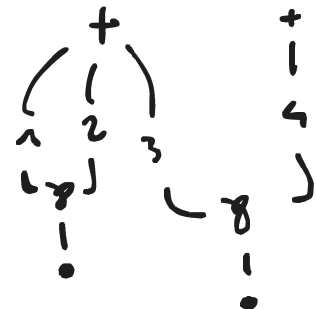
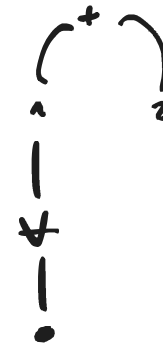
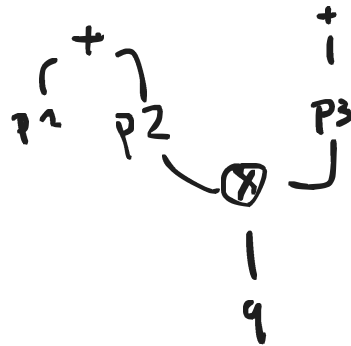
$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:



examples



TYPED VS UNTYPED

1 - Untyped nets (naive)

2 - How to type a net

3 - is atomic typing enough?

4 - typing is non-trivial

5 - Comparative

6 - Parsing as type inference

7 - More incorrectness

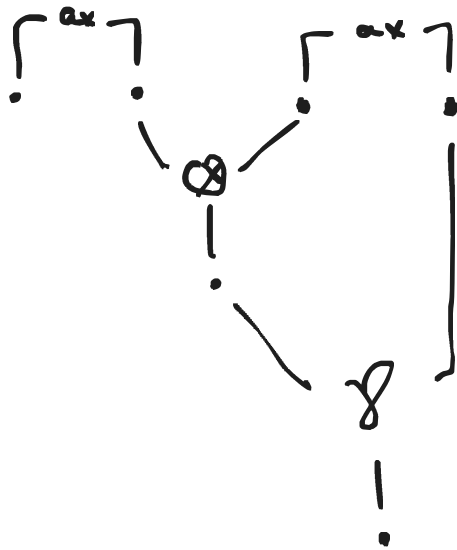
How to type a net?

How to type a net?

γ : Untyped net $\mapsto$ typed net

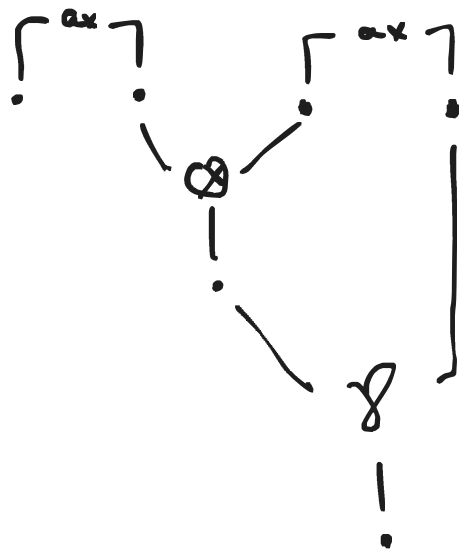
How to type a net?

γ : Untyped net $\mapsto$ typed net

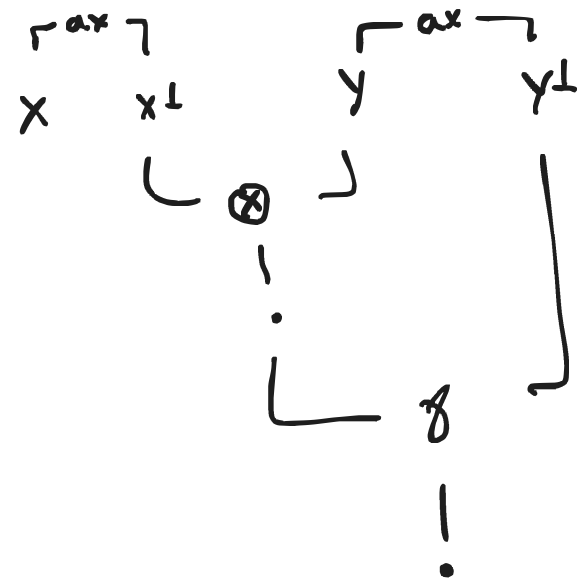


How to type a net?

γ : Untyped net $\mapsto$ typed net

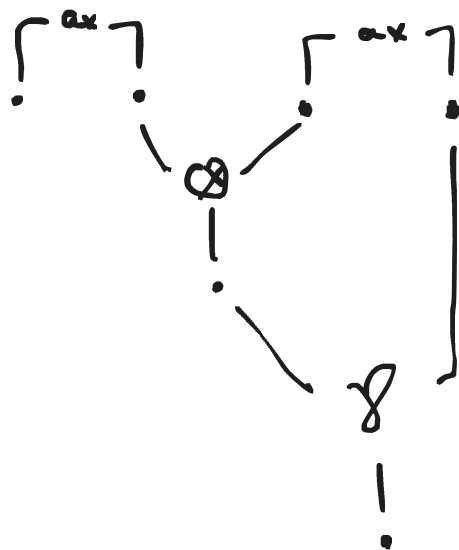


$\mapsto$

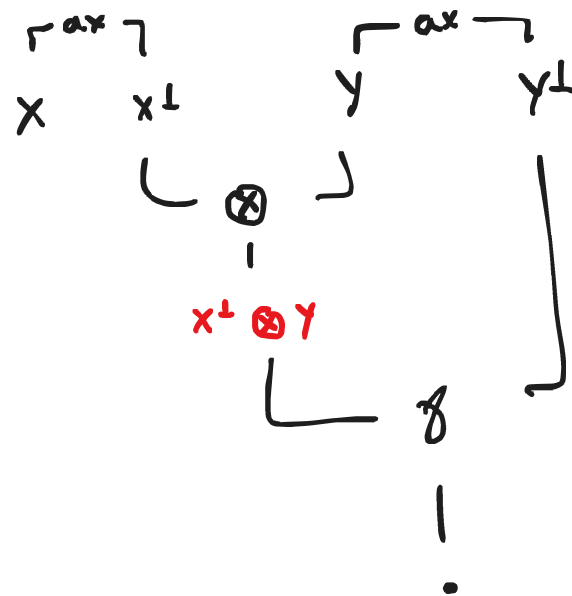


How to type a net?

γ : Untyped net $\mapsto$ typed net

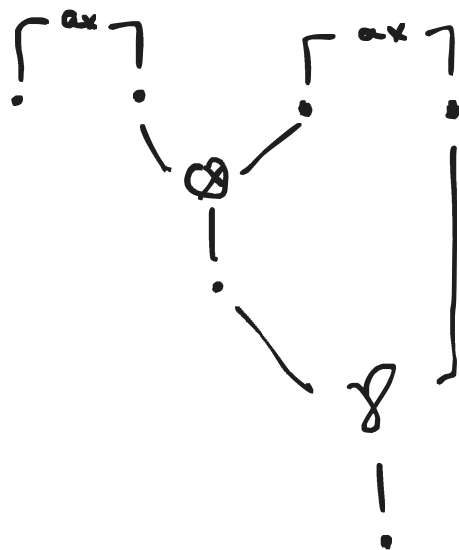


$\mapsto$

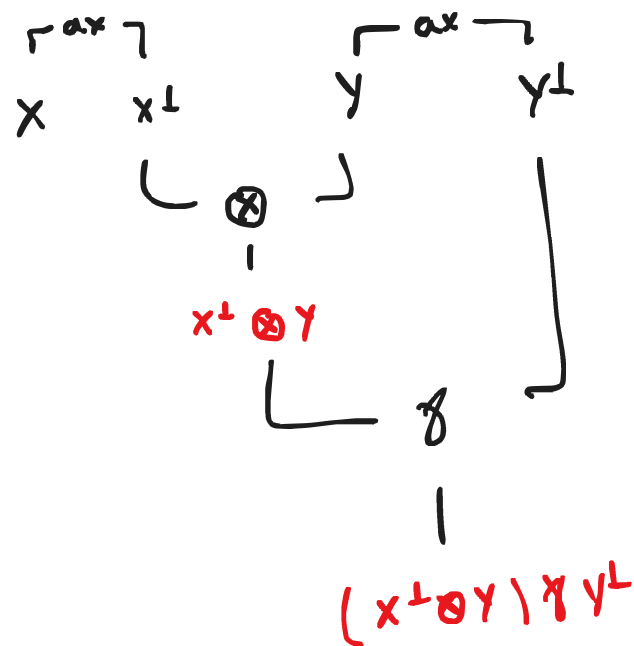


How to type a net?

γ : Untyped net $\mapsto$ typed net



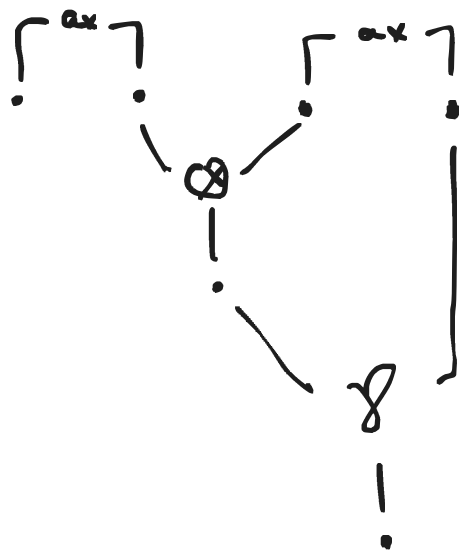
$\mapsto$



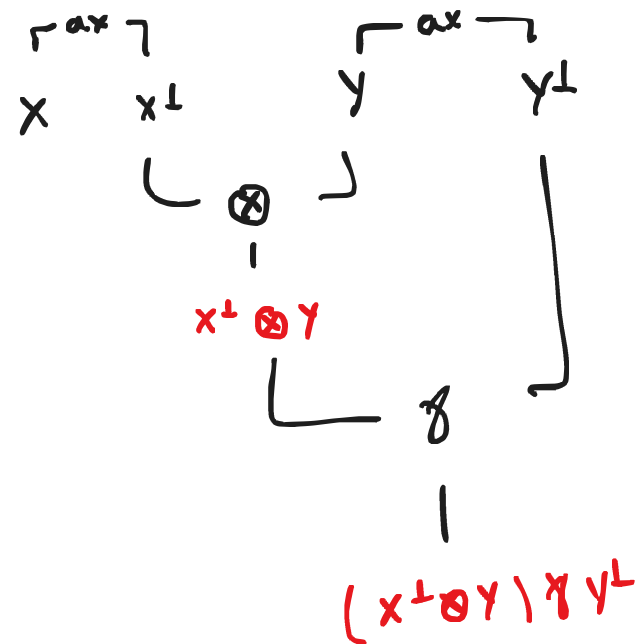
How to type a net?

γ : Untyped net $\mapsto$ typed net

OK without quantifiers!



$\mapsto$

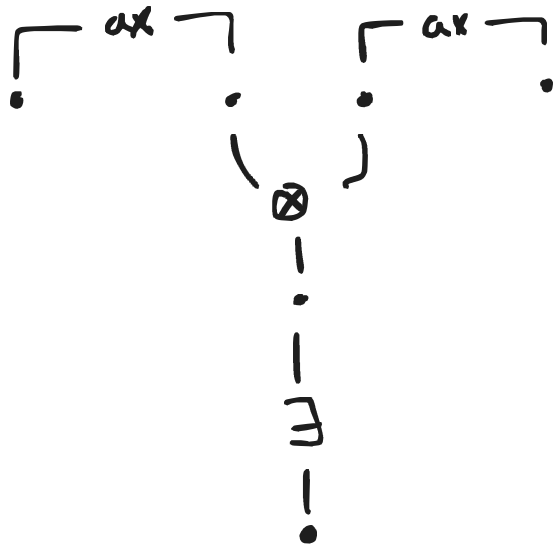


How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

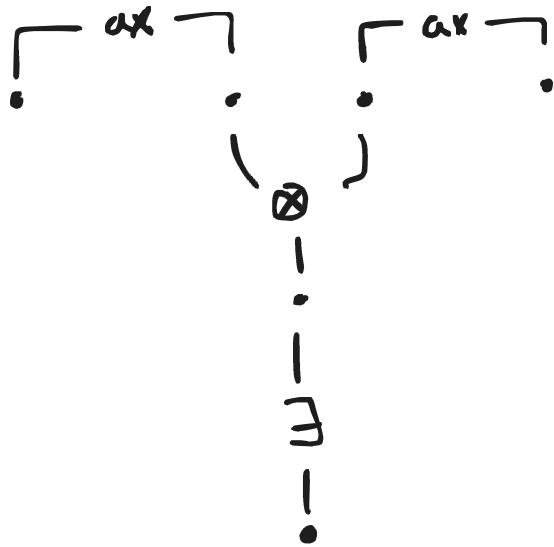
How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

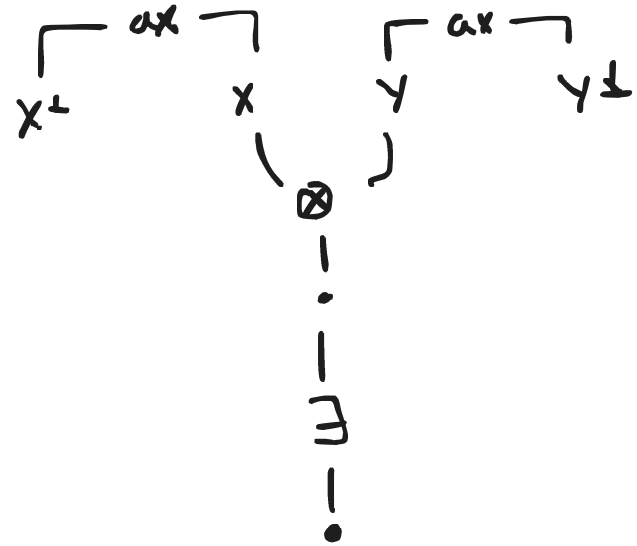


How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

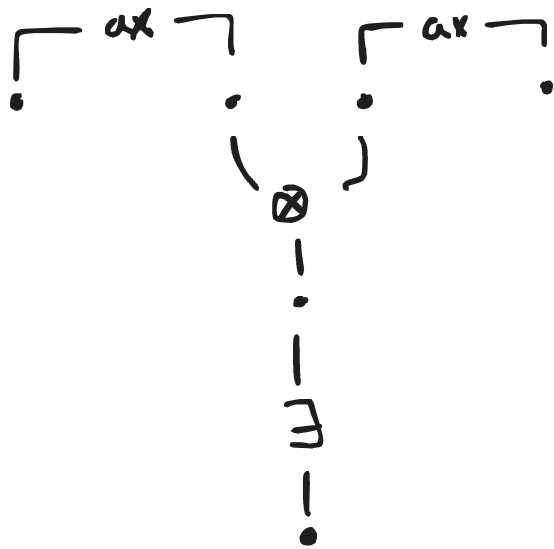


$\mapsto$

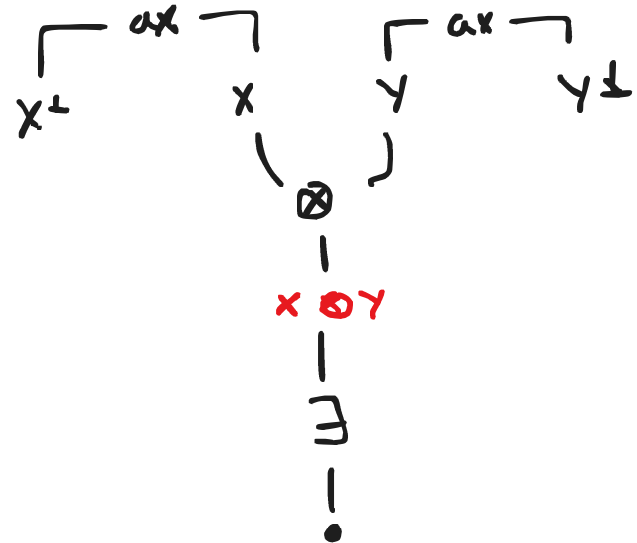


How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

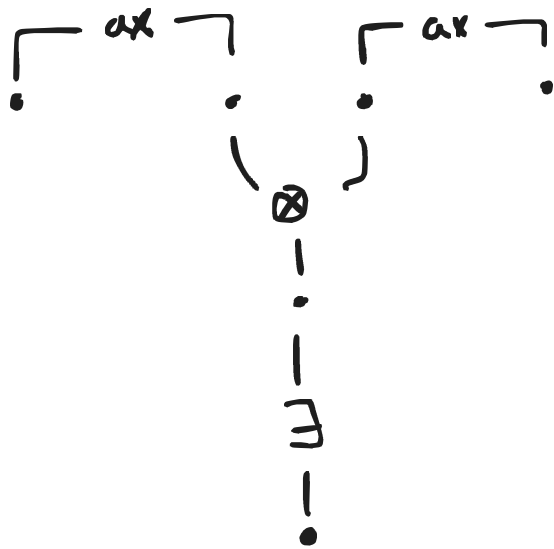


$\mapsto$

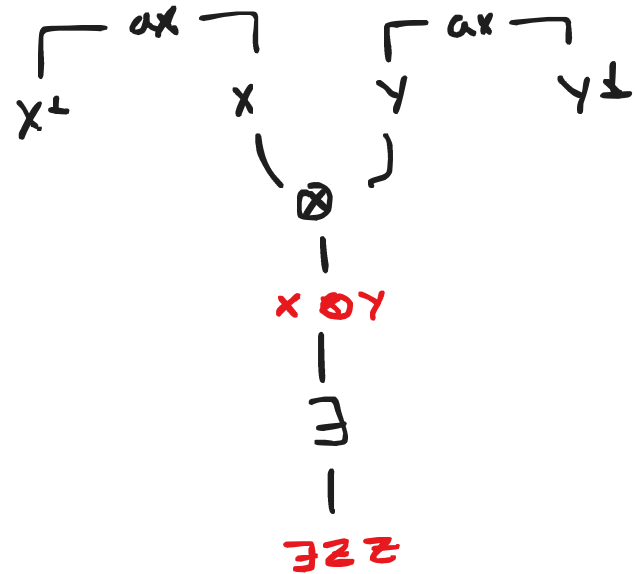


How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

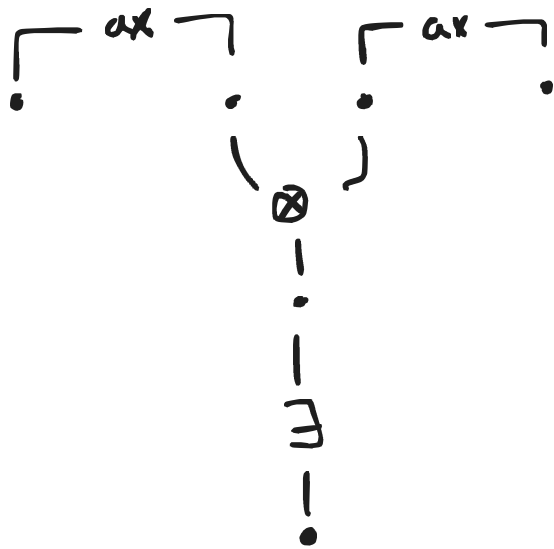


$\mapsto$

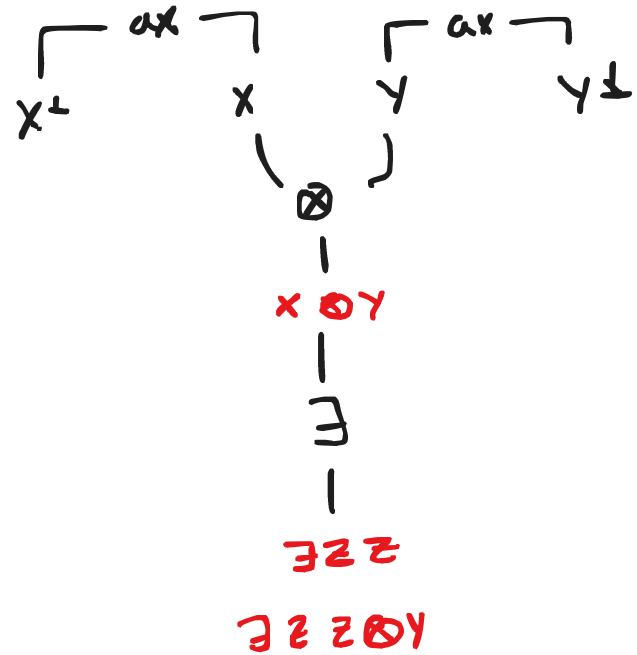


How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

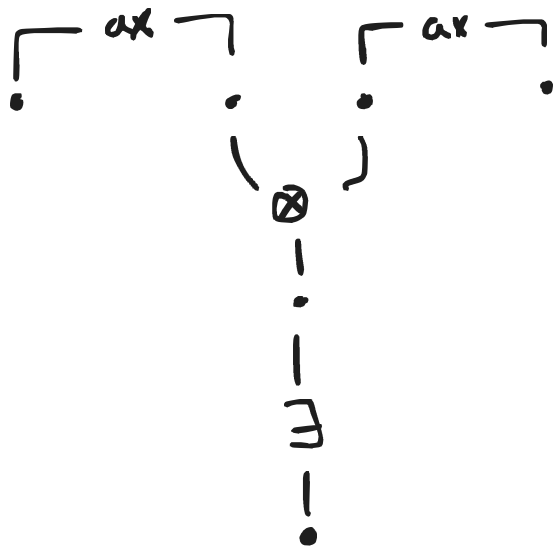


$\mapsto$

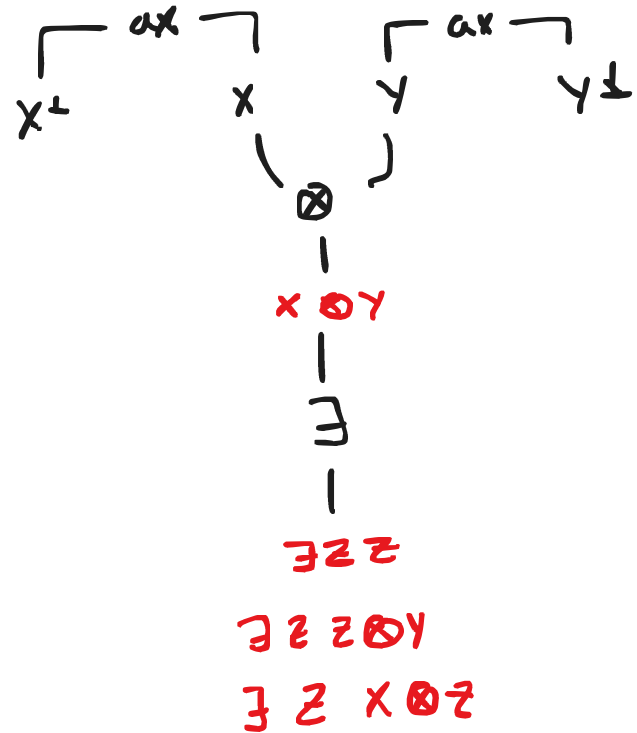


How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

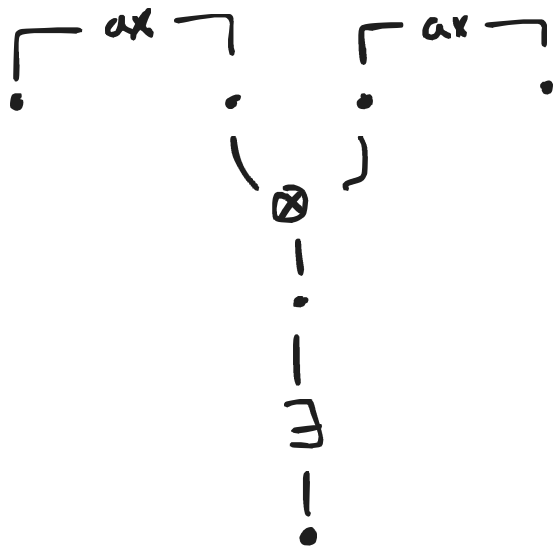


$\mapsto$

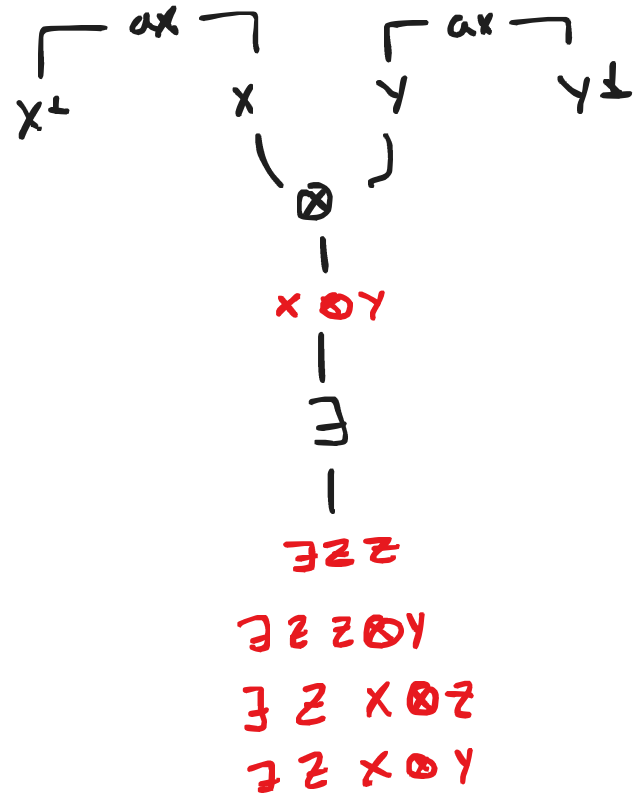


How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

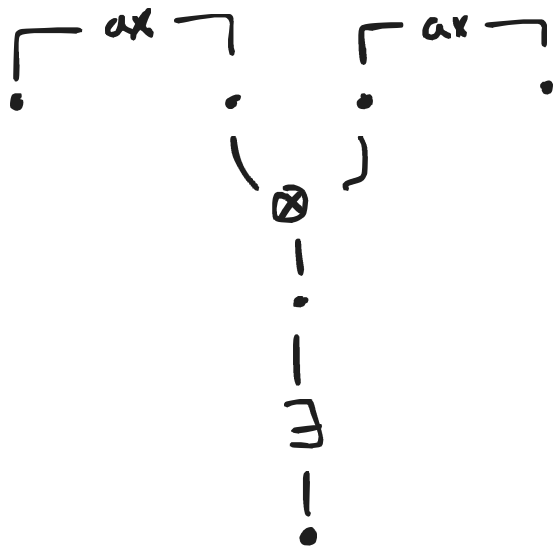


$\mapsto$

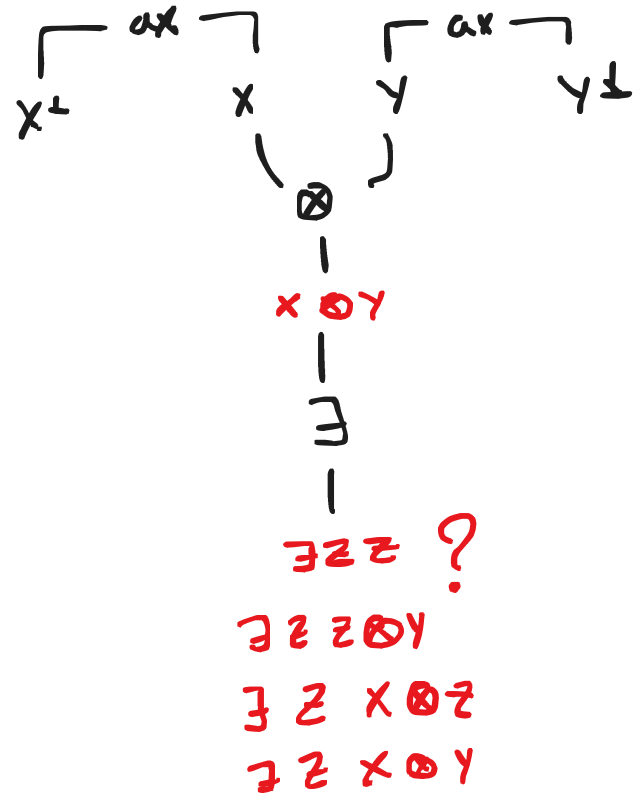


How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)



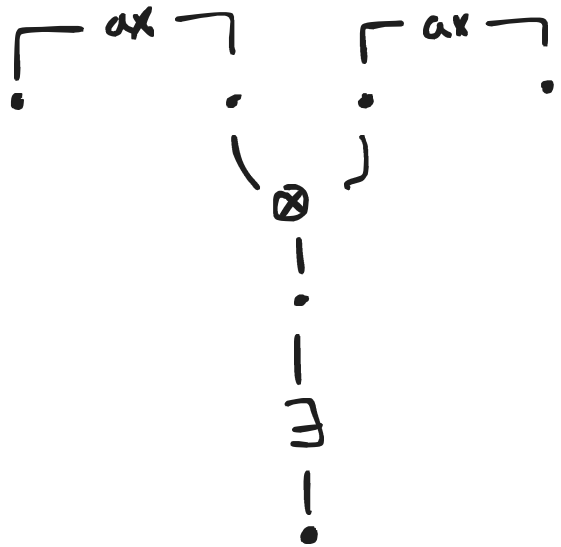
$\mapsto$



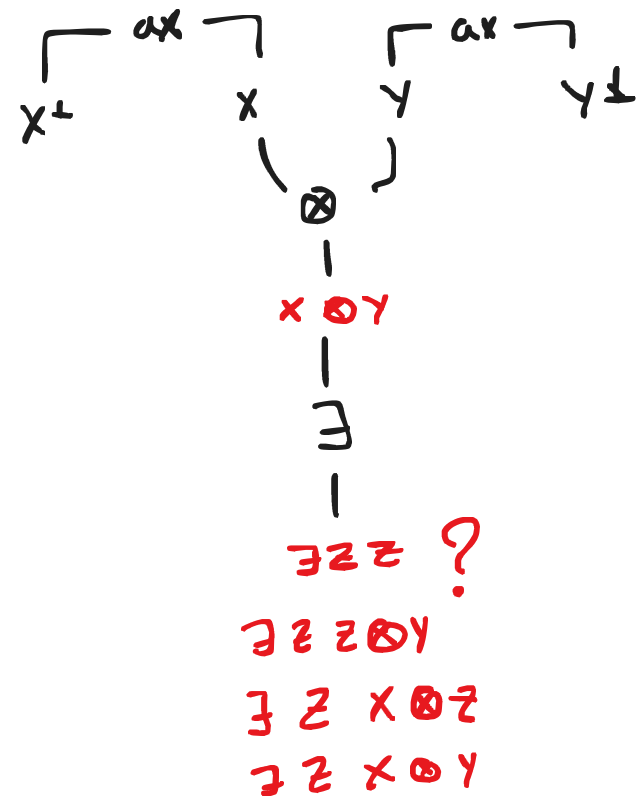
How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

$\rightarrow$ 3-pointers



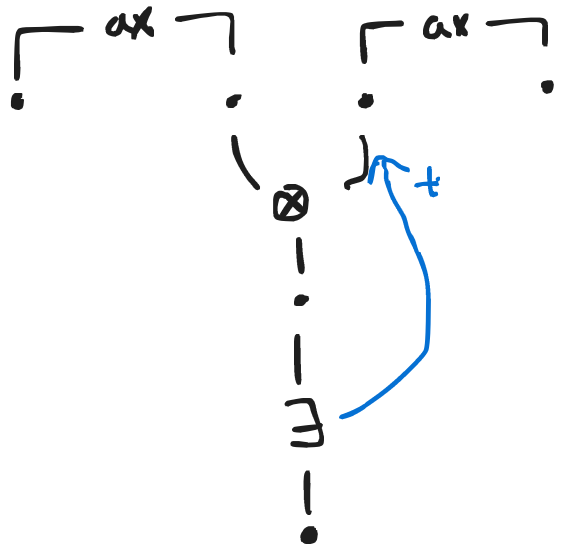
$\mapsto$



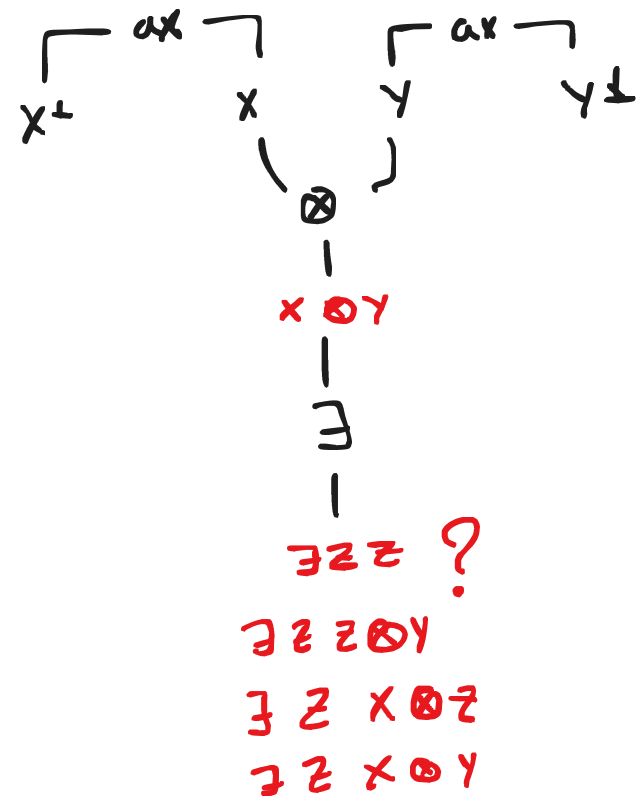
How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

$\rightarrow \exists$ -pointers



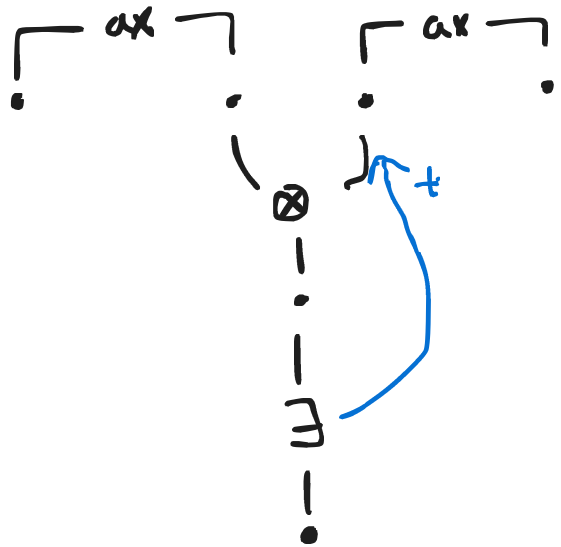
$\mapsto$



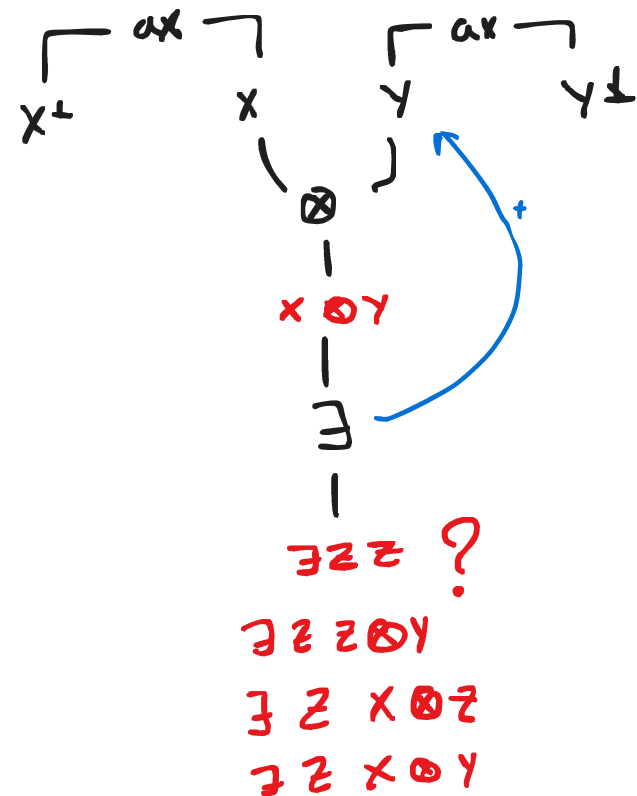
How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

$\rightarrow$ 3-pointers



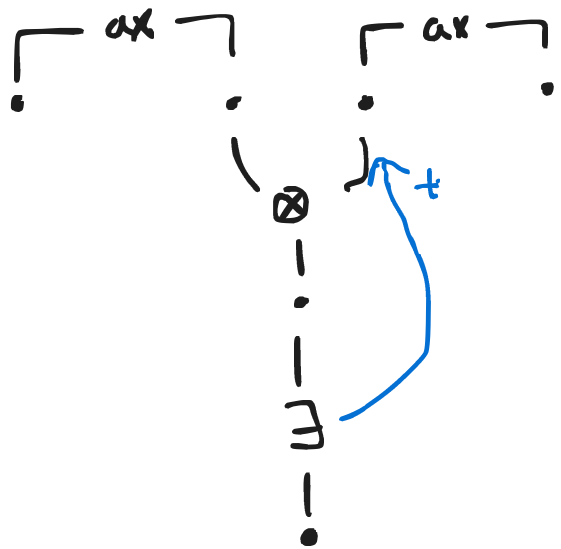
$\mapsto$



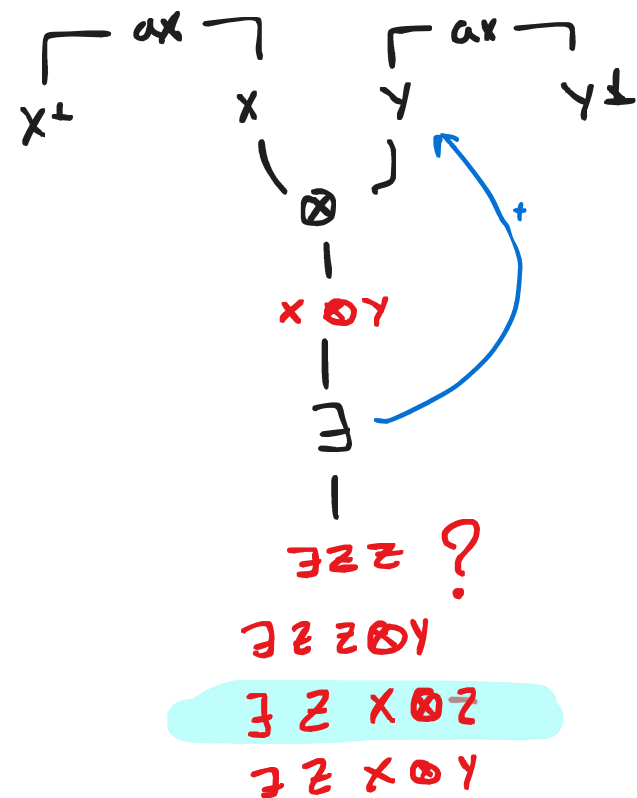
How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

$\rightarrow$ 3-pointers



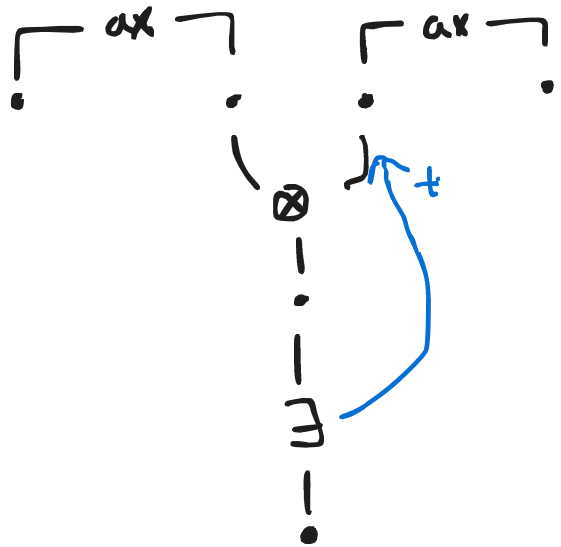
$\mapsto$



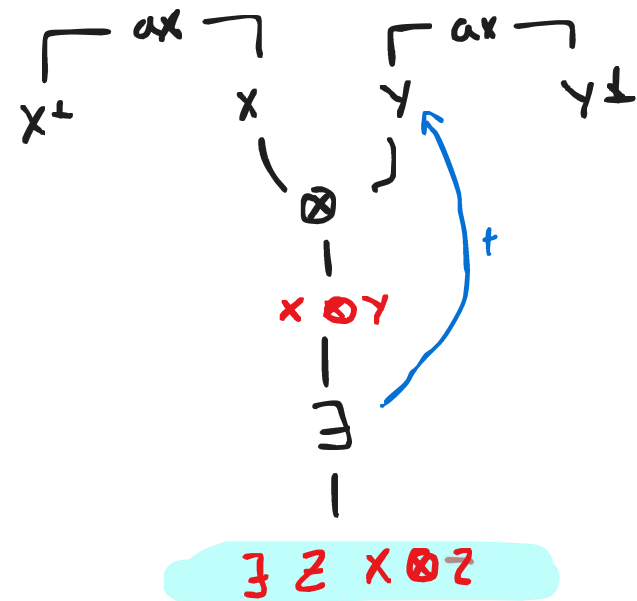
How to type a net?

Untyped net $\mapsto$ typed net (with quantifiers)

$\rightarrow \exists$ -pointers



$\mapsto$



TYPED VS UNTYPED

- 1 - Untyped nets (naive)
- 2 - How to type a net
- 3 - is atomic typing enough?
- 4 - typing is non-trivial
- 5 - Comparative
- 6 - Parsing as type inference
- 7 - More incorrectness

is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for η -expanded typed nets. solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for η -expanded typed nets. solved

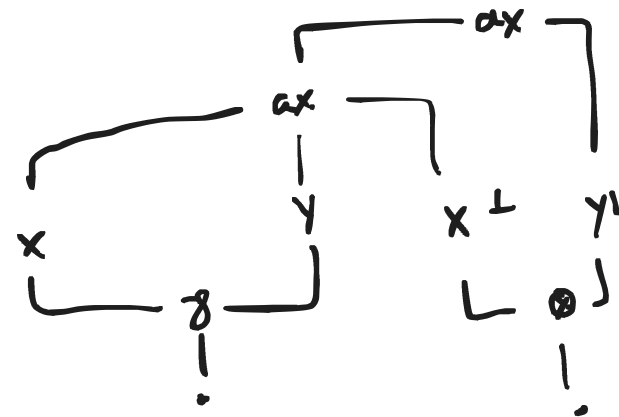
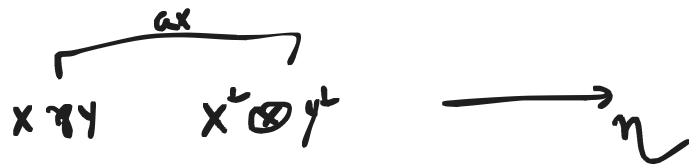
$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

$$\frac{x \quad y}{x^{\perp} \otimes y^{\perp}} \alpha x$$

is atomic Typing enough?

Hypothesis: problem **P** for η -expanded typed nets. solved

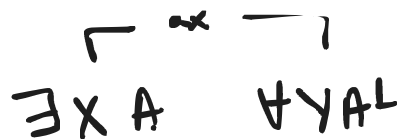
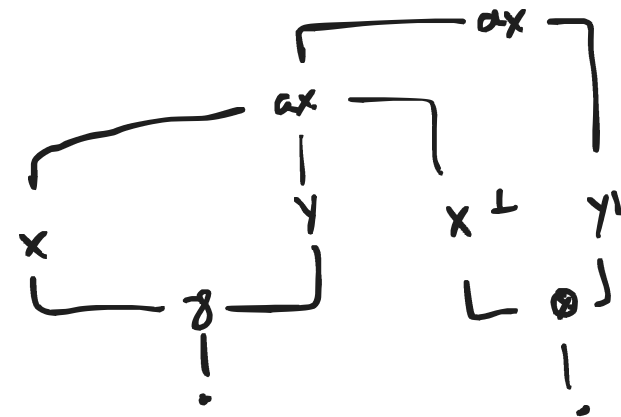
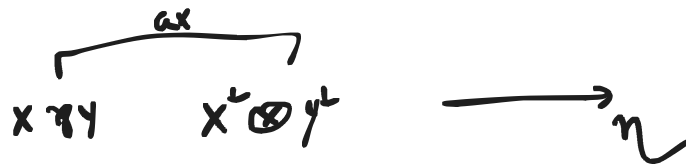
$\hookrightarrow$ can we solve **P** for any typed nets?



is atomic Typing enough?

Hypothesis: problem **P** for η -expanded typed nets. solved

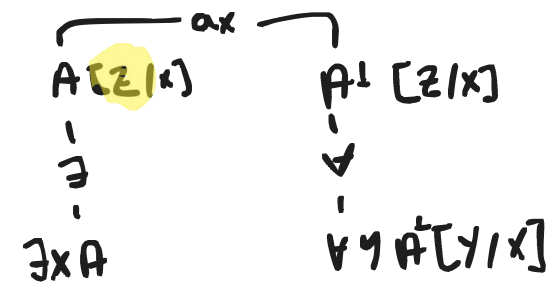
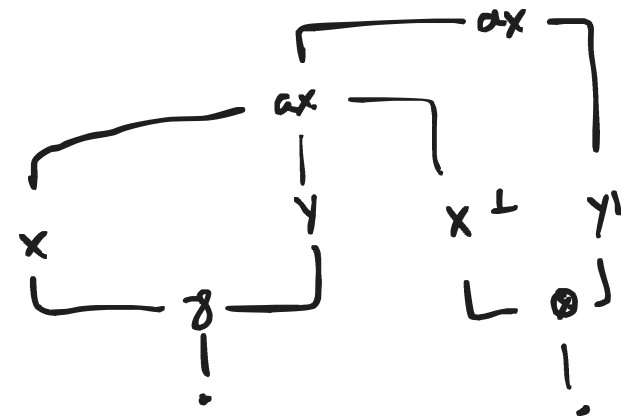
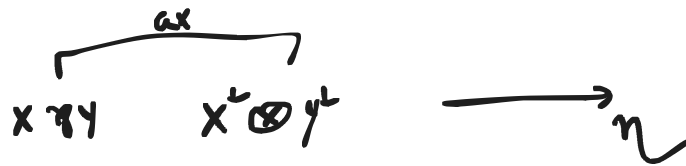
$\hookrightarrow$ can we solve **P** for any typed nets?



is atomic Typing enough?

Hypothesis: problem **P** for η -expanded typed nets. solved

↳ can we solve **P** for any typed nets?

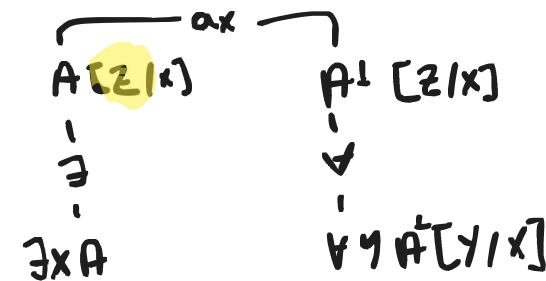
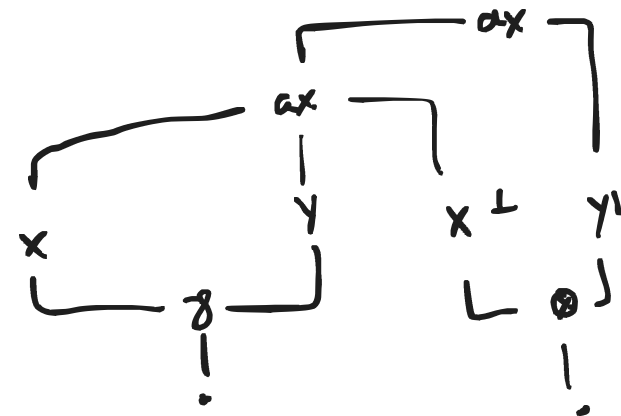
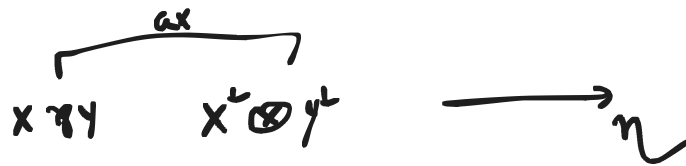


is atomic Typing enough?

Hypothesis: problem **P** for η -expanded typed nets. solved

$\hookrightarrow$ can we solve **P** for any typed nets?

Yes!

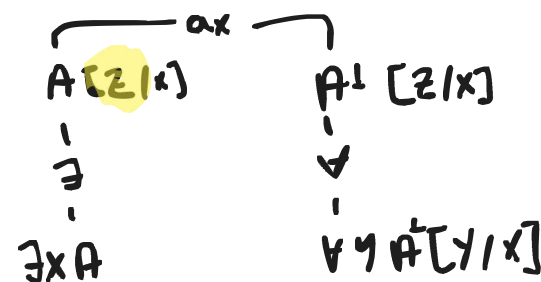
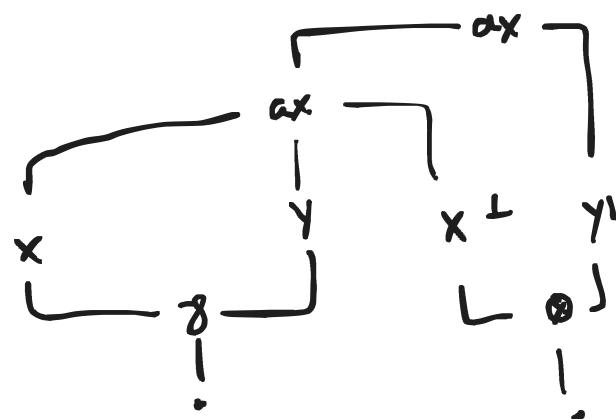
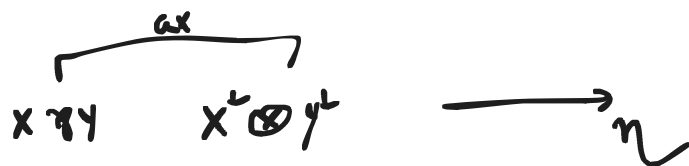


is atomic Typing enough?

Hypothesis: problem **P** for η -expanded typed nets. solved

↳ can we solve **P** for any typed nets?

Yes! (w/o daimons!)



is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for η -expanded typed nets. solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

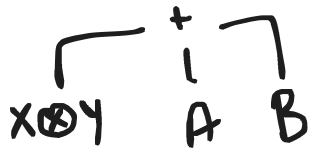
With
deimons?

is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets. solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

With
daimons?

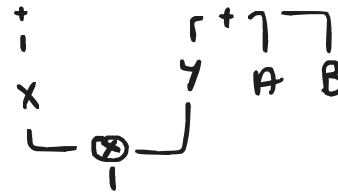
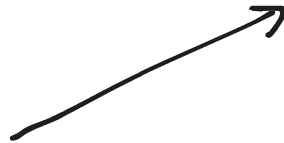
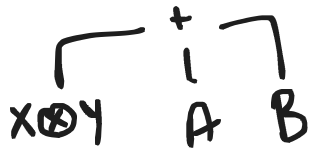


is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets. solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

With
daimons?

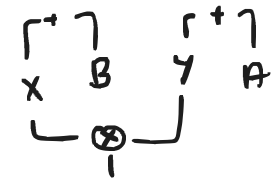
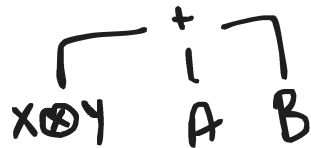


is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets. solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

With
dimons?

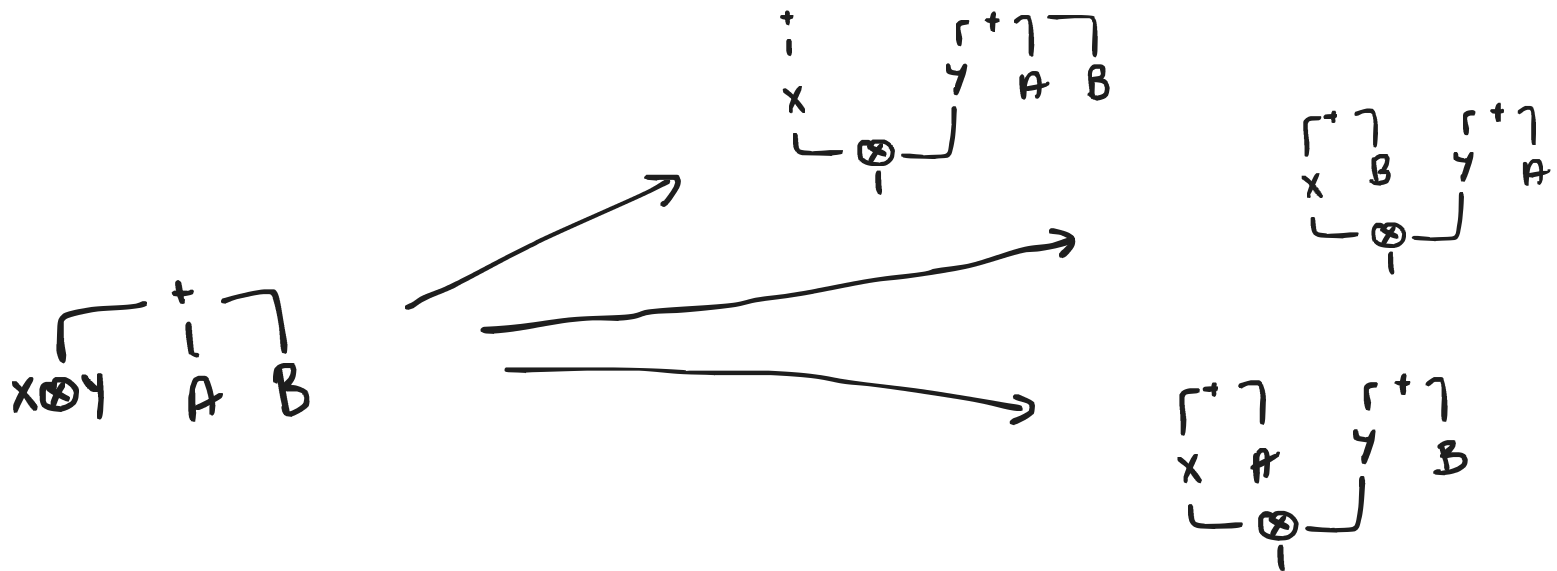


is atomic Typing enough?

Hypothesis: problem **P** for **atomically** typed nets. solved

↳ can we solve **P** for any typed nets?

With
dimons?

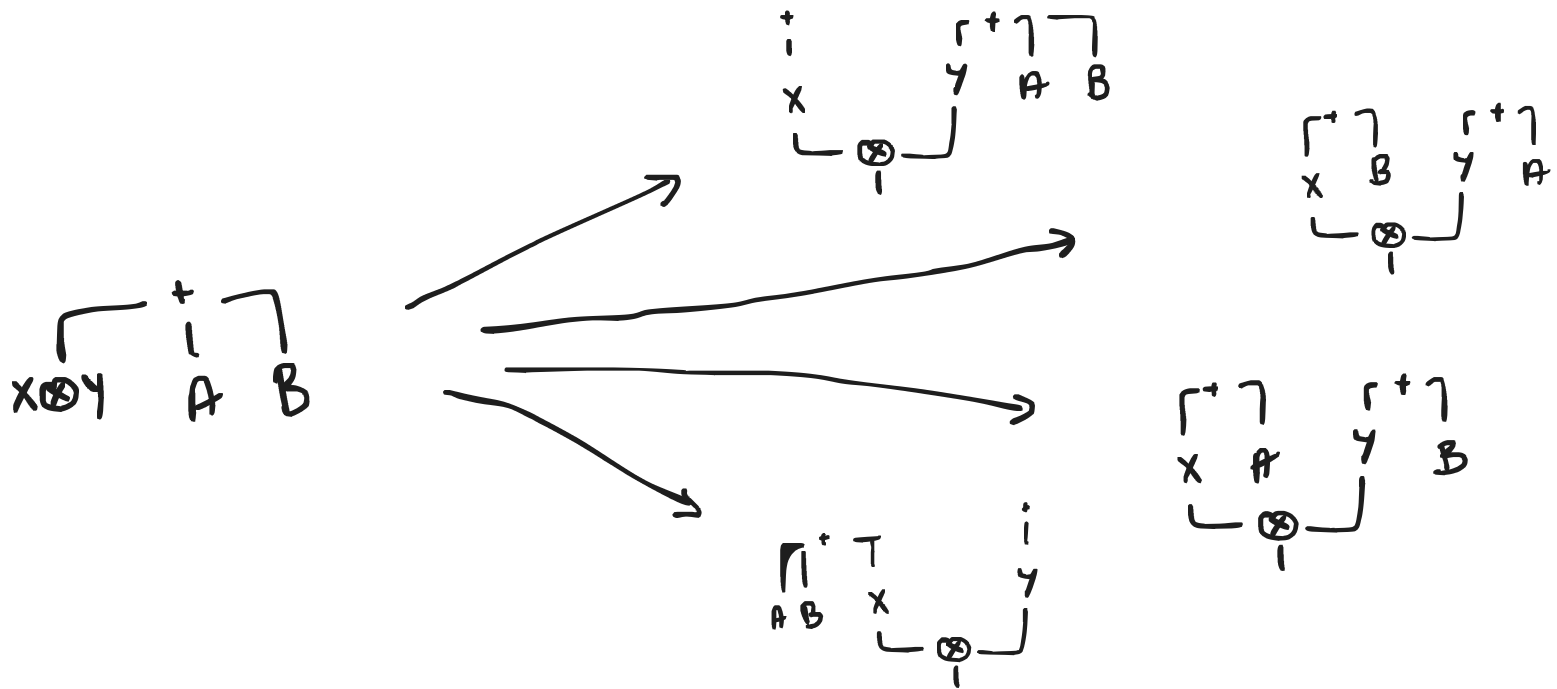


is atomic Typing enough?

Hypothesis: problem **P** for **atomically** typed nets. solved

↳ can we solve **P** for any typed nets?

With
daimons?



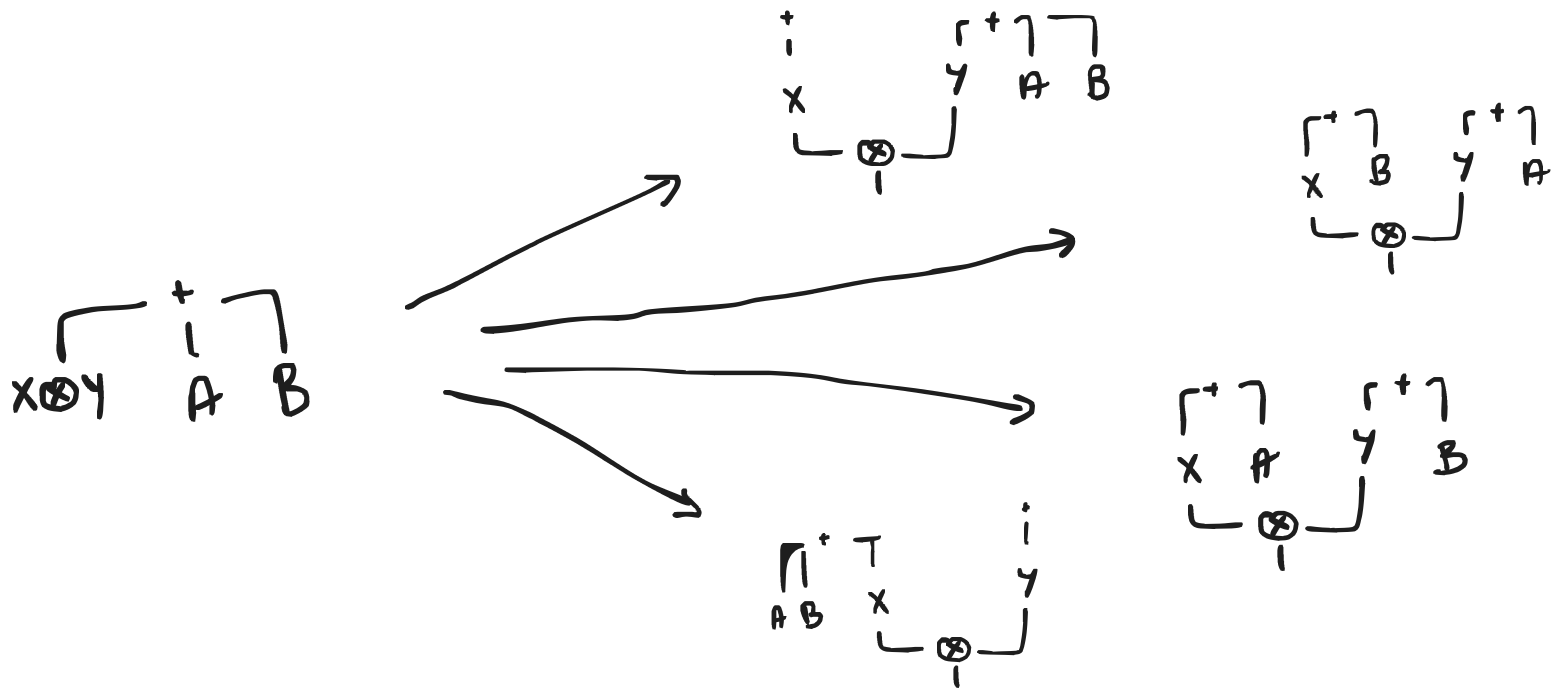
is atomic Typing enough?

Hypothesis: problem **P** for **atomically** typed nets. solved

↳ can we solve **P** for any typed nets?

Yes but
EXPENSIVE!

With
deimons?



is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

With
deimons + quantifiers?

is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

With
deimons + quantifiers?

+

$\exists x A(x, z)$

is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

With
atoms + quantifiers?

$$\begin{array}{c} + \\ | \\ \exists x A[x/z] \end{array}$$

$\rightarrow$

$$\begin{array}{c} + \\ | \\ A[B/z] \\ | \\ \exists \\ | \\ \exists x A[x/z] \end{array}$$

is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

With
atoms + quantifiers?

$$\begin{array}{c} + \\ | \\ \exists x A[x/z] \end{array}$$

$\rightarrow$

$$\begin{array}{c} + \\ | \\ A[B/z] \\ | \\ \exists \\ | \\ \exists x A[x/z] \end{array} \quad \text{any } B?$$

is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

With
atoms + quantifiers?

$$\begin{array}{c} + \\ | \\ \exists x A[x/z] \end{array}$$

$\rightarrow$

$$\begin{array}{c} + \\ | \\ A[B/z] \\ | \\ \exists \\ | \\ \exists x A[x/z] \end{array}$$

any B?

B may contain $\exists$

is atomic Typing enough?

Hypothesis: problem $\mathcal{P}$ for atomically typed nets solved

$\hookrightarrow$ can we solve $\mathcal{P}$ for any typed nets?

No

With
disjoints + quantifiers?

$\vdash$
 $\exists x A[x/z]$

$\rightarrow$

$\vdash$
 $A[B/z]$
 $\vdash$
 $\exists x A[x/z]$

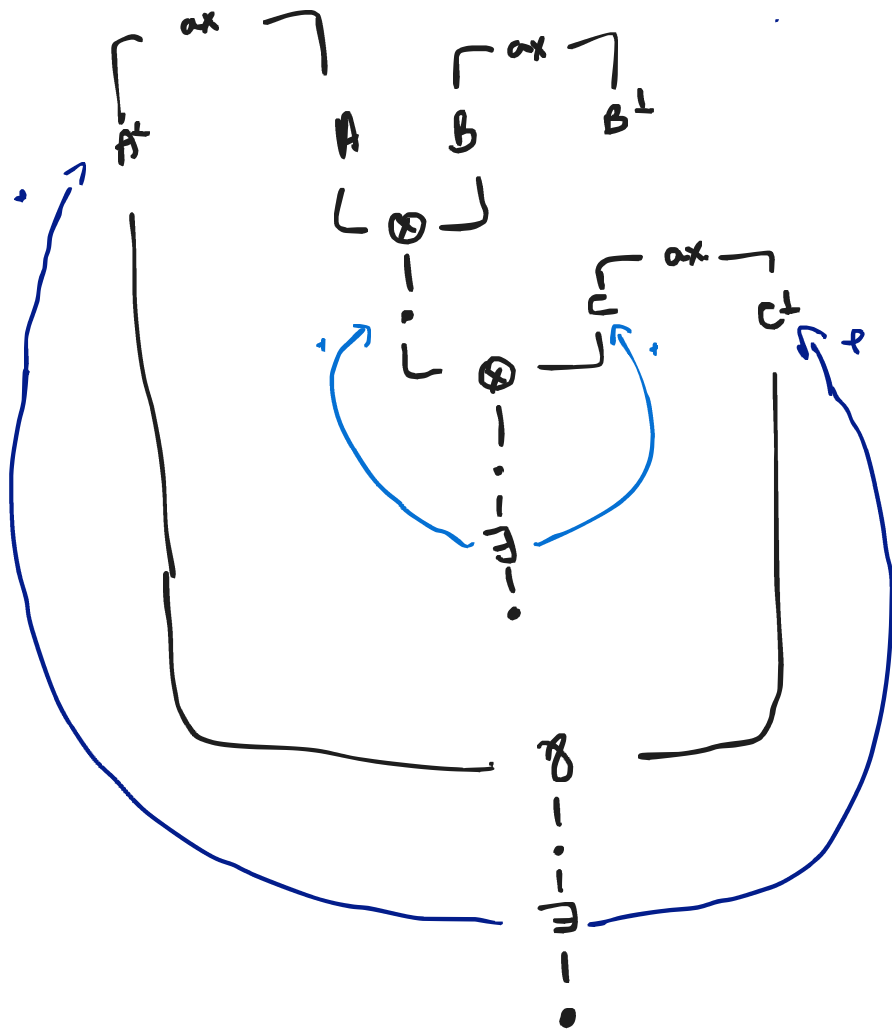
any B?

B may contain $\exists$

TYPED VS UNTYPED

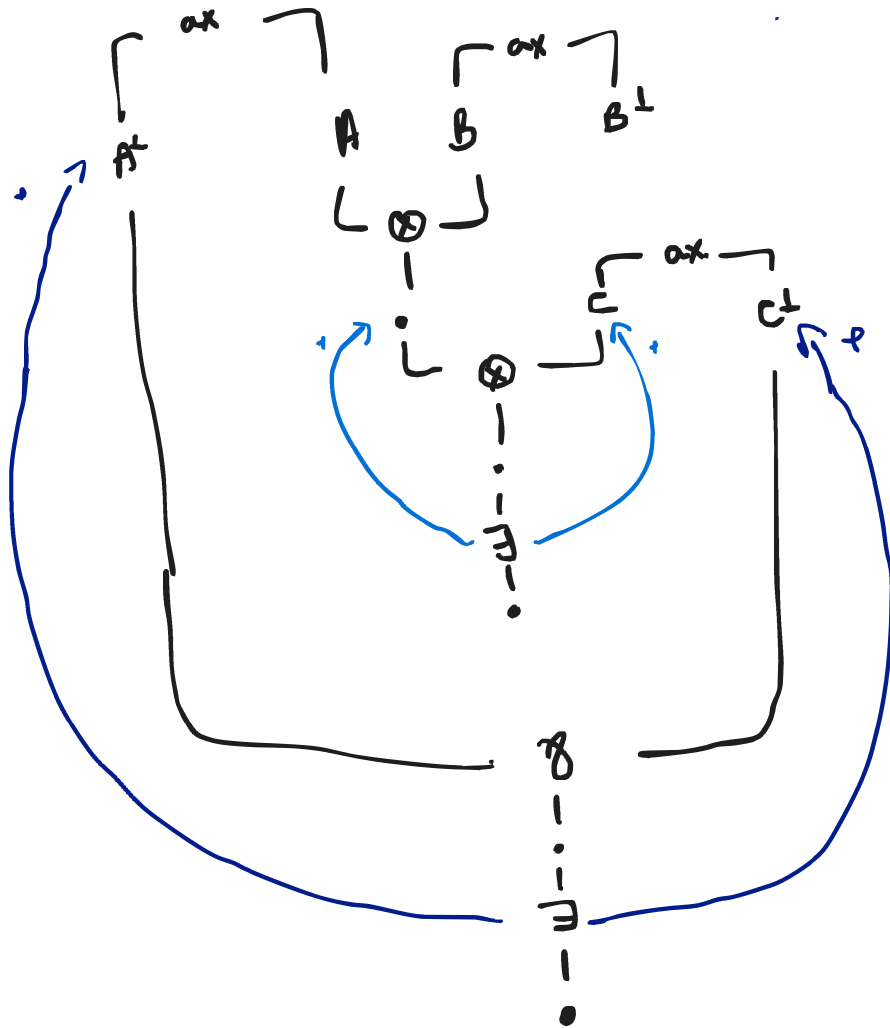
- 1 - Untyped nets (naive)
- 2 - How to type a net
- 3 - is atomic typing enough?
- 4 - typing is non-trivial
- 5 - Comparative
- 6 - Parsing as type inference
- 7 - More incorrectness

Typing is non-trivial

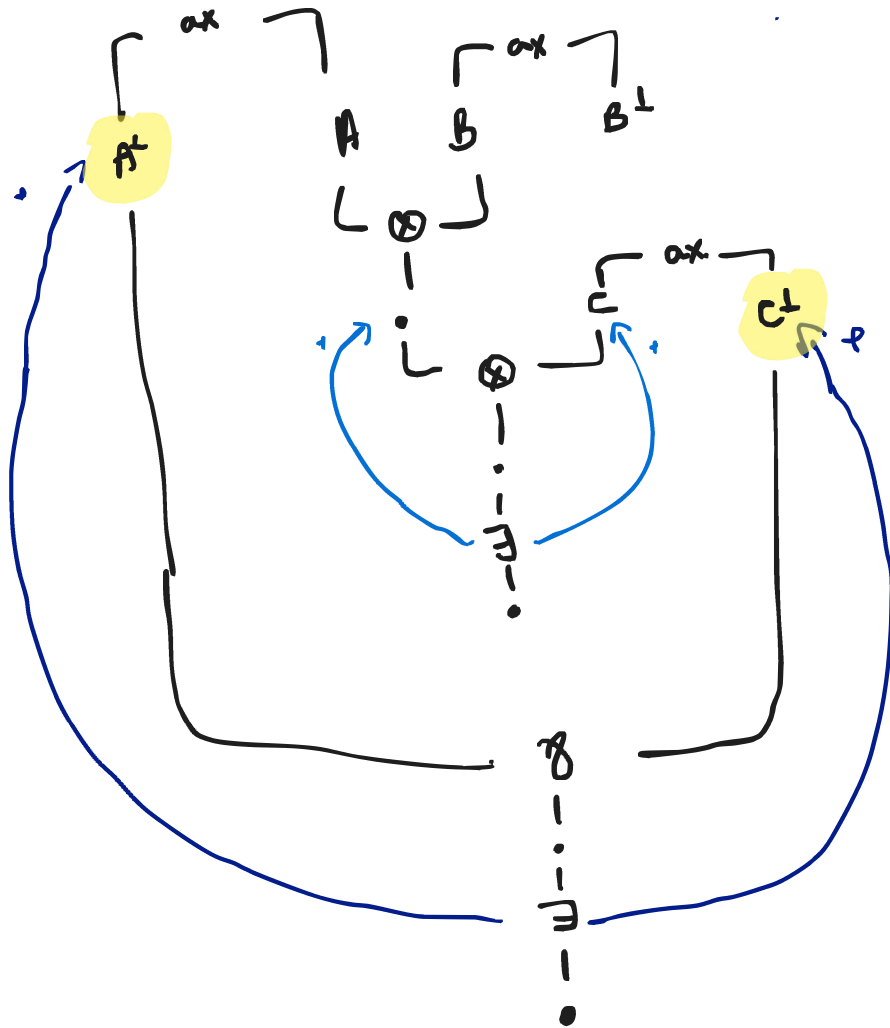


Typing is non-trivial

CONSTRAINTS



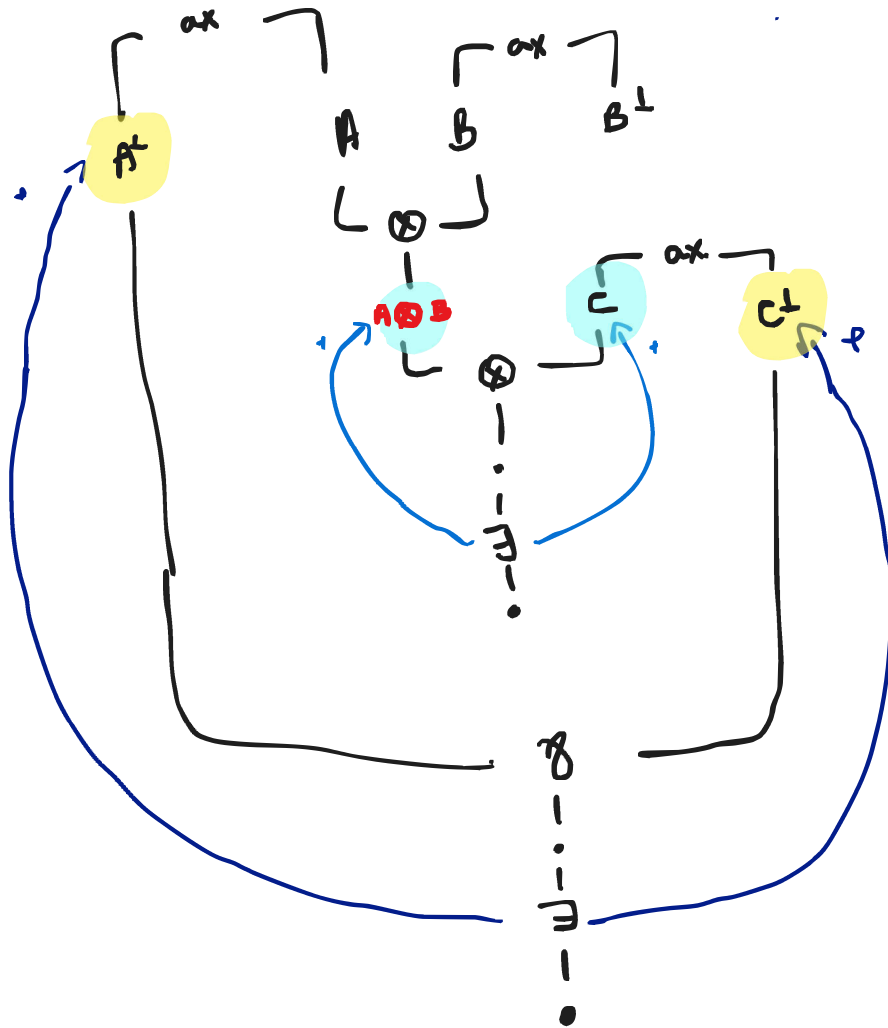
Typing is non-trivial



CONSTRAINTS

1- $A^\perp = C^\perp \iff A = C$

Typing is non-trivial

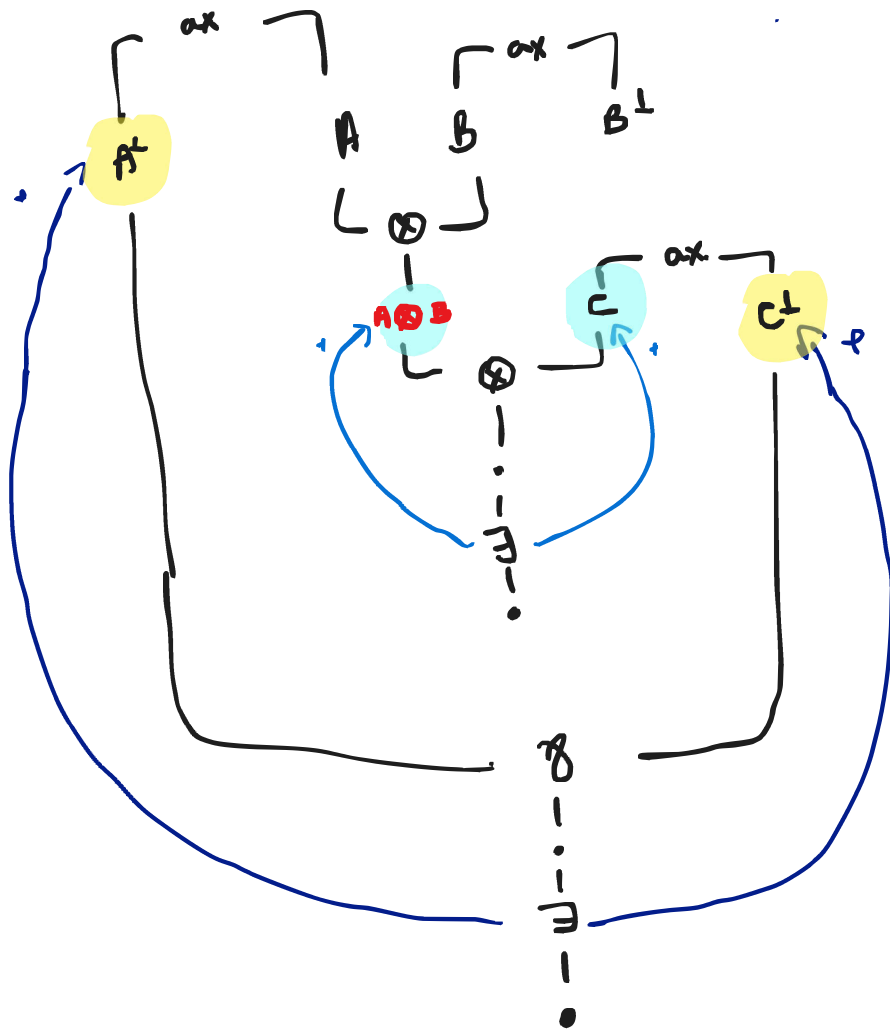


CONSTRAINTS

1- $A^\perp = C^\perp \iff A = C$

2- $C = A \otimes B$

Typing is non-trivial



CONSTRAINTS

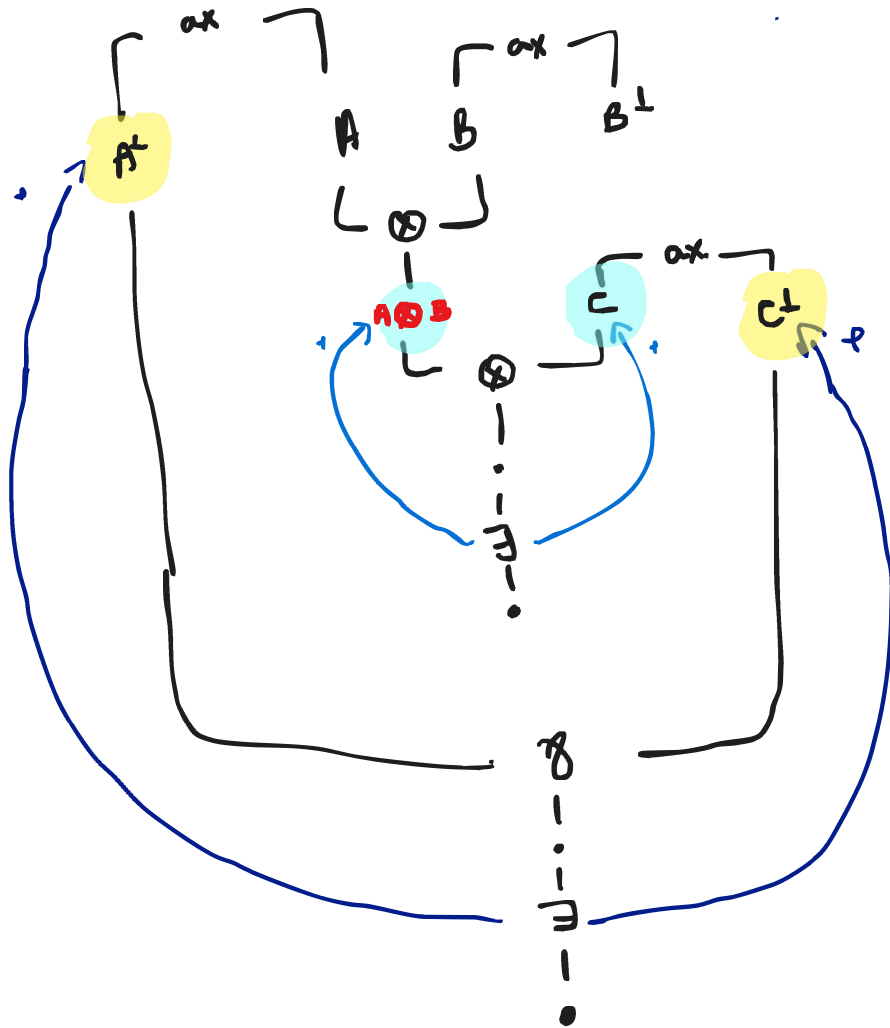
2- $A^\perp = C^\perp \Leftrightarrow A = C$

2- $C = A \otimes B$

$$A = A \otimes B$$

impossible!

Typing is non-trivial



CONSTRAINTS

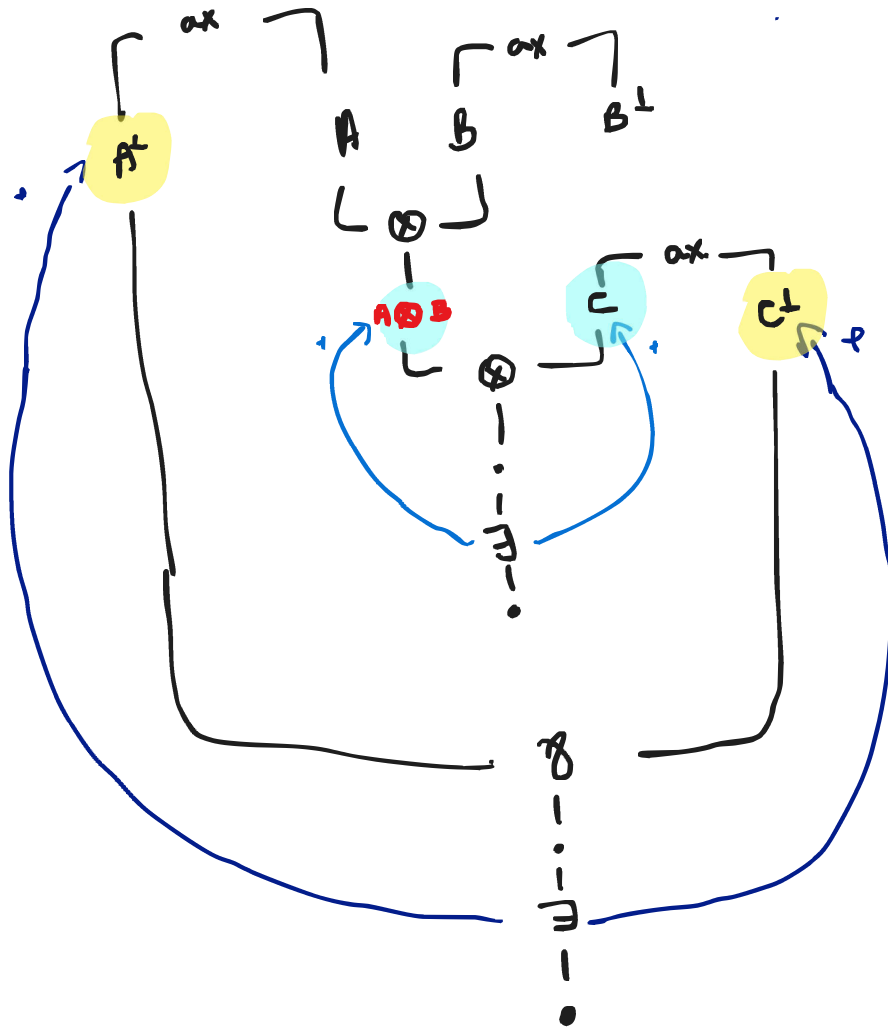
4- $A^\perp = C^\perp \Leftrightarrow A = C$

2- $C = A \otimes B$

$$A = A \otimes B$$

impossible!

Typing is non-trivial



CONSTRAINTS

1- $A^{\perp} = C^{\perp} \iff A = C$

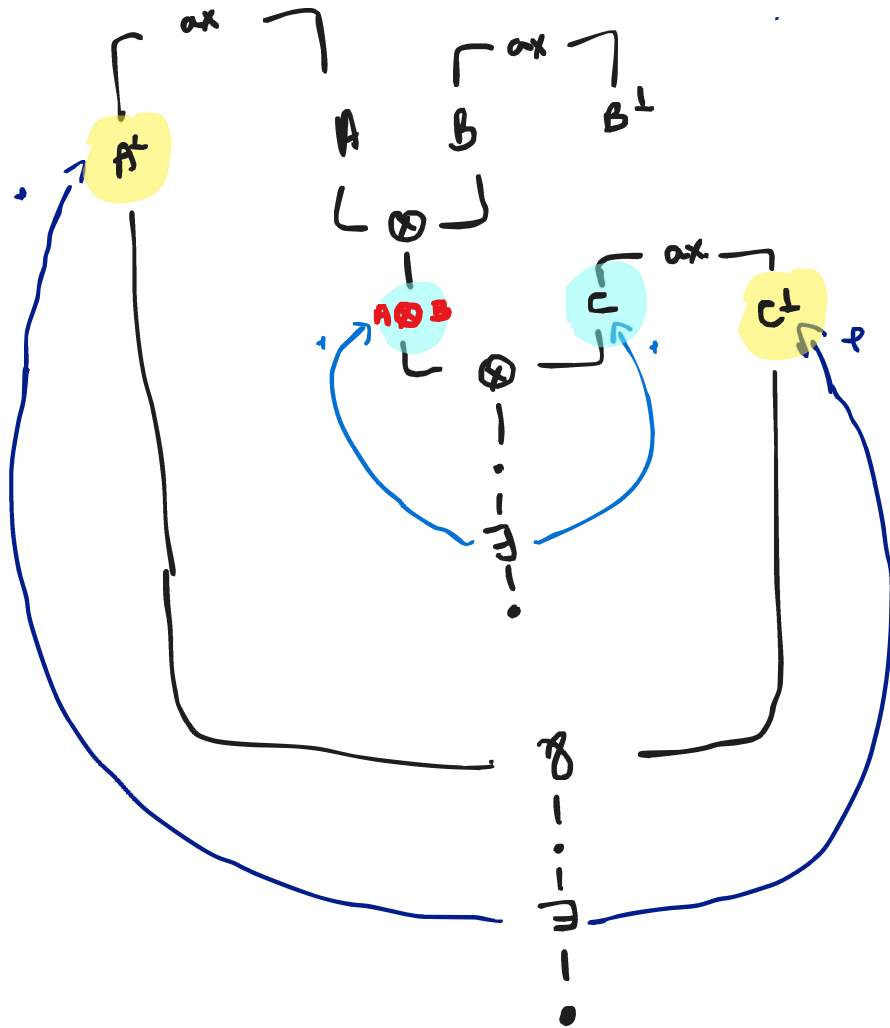
2- $C = A \otimes B$

$A = A \otimes B$

impossible!

$$\frac{\overline{A, A^{\perp}} \quad \overline{B, B^{\perp}}}{A \otimes B, A^{\perp}, B^{\perp}} \textcircled{2}$$

Typing is non-trivial



CONSTRAINTS

1- $A^\perp = C^\perp \iff A = C$

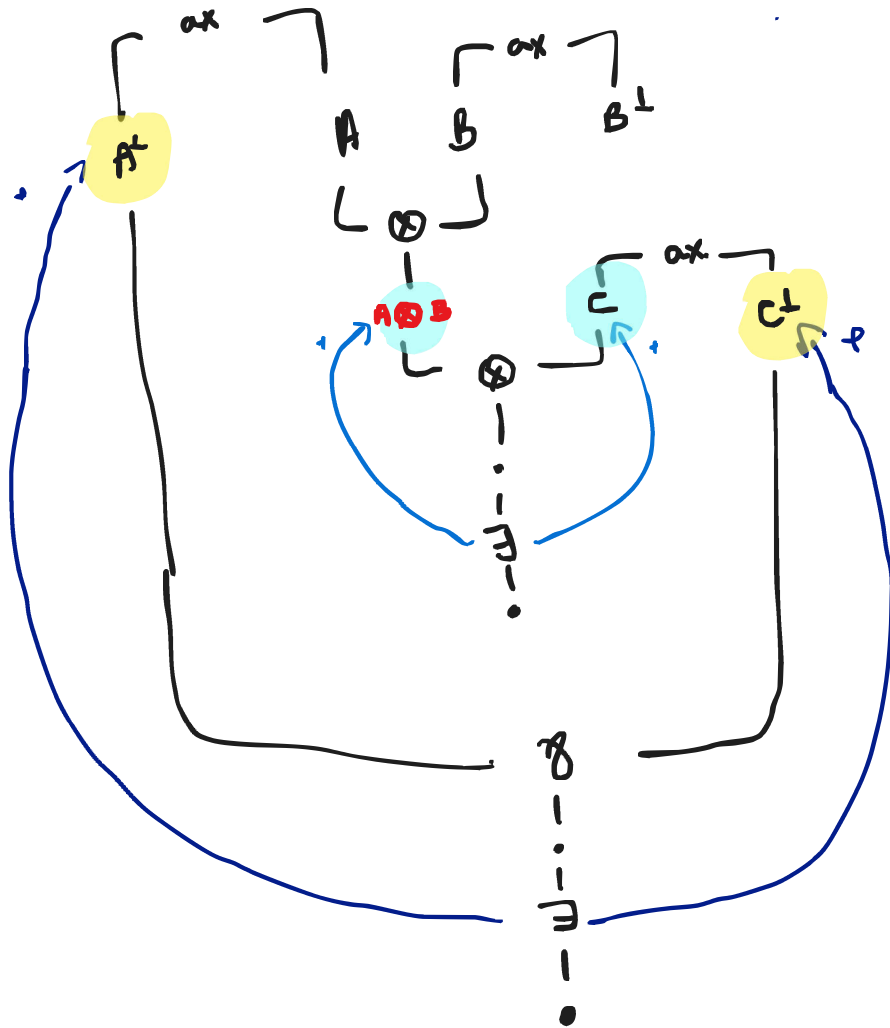
2- $C = A \otimes B$

$A = A \otimes B$

impossible!

$$\frac{\frac{\overline{A, A^\perp} \quad \overline{B, B^\perp}}{A \otimes B, A^\perp, B^\perp} \otimes \quad \overline{C, C^\perp}}{A \otimes B \otimes C, A^\perp, B^\perp, C^\perp} \otimes$$

Typing is non-trivial



CONSTRAINTS

1- $A^L = C^L \iff A = C$

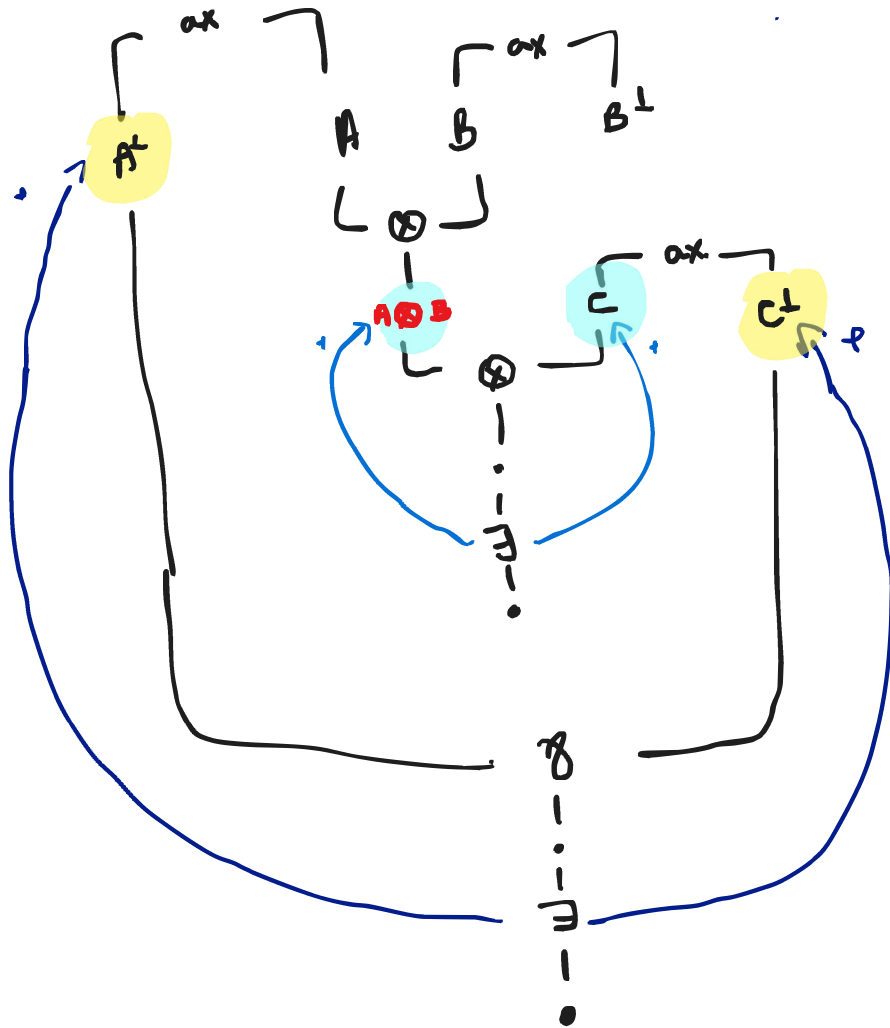
2- $C = A \otimes B$

$A = A \otimes B$

impossible!

$$\begin{array}{c}
 \frac{\overline{A, A^L} \quad \overline{B, B^L}}{A \otimes B, A^L, B^L} \otimes \\
 \frac{\quad \overline{C, C^L}}{A \otimes B \otimes C, A^L, B^L, C^L} \otimes \\
 \hline
 \exists z \exists z \otimes z, A^L, B^L, C^L
 \end{array}$$

Typing is non-trivial



CONSTRAINTS

1- $A^{\perp} = C^{\perp} \iff A = C$

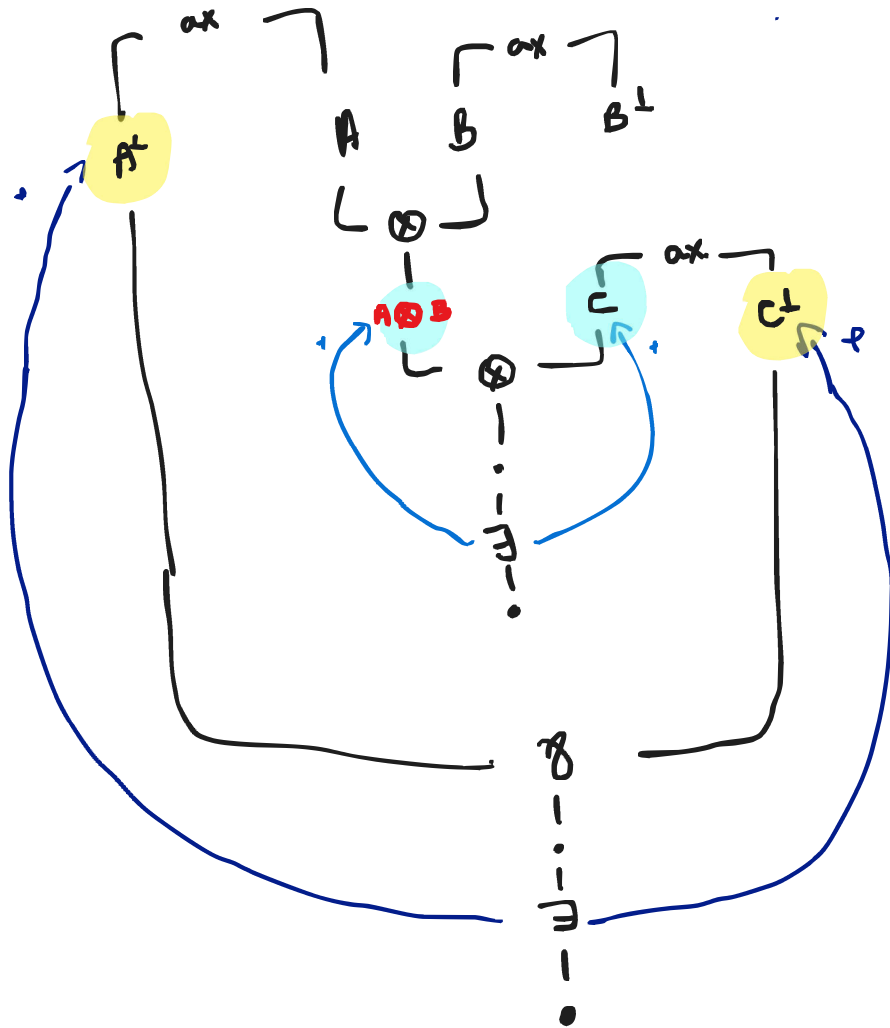
2- $C = A \otimes B$

$A = A \otimes B$

impossible!

$$\begin{array}{c}
 \frac{\overline{A, A^{\perp}} \quad \overline{B, B^{\perp}}}{A \otimes B, A^{\perp}, B^{\perp}} \otimes \\
 \frac{A \otimes B, A^{\perp}, B^{\perp} \quad A \otimes B, A^{\perp} \otimes B^{\perp}}{A \otimes B \otimes (A \otimes B), A^{\perp}, B^{\perp}, A^{\perp} \otimes B^{\perp}} \otimes \\
 \hline
 \exists z \exists z \otimes z, A^{\perp}, B^{\perp}, A^{\perp} \otimes B^{\perp}
 \end{array}$$

Typing is non-trivial



CONSTRAINTS

1- $A^\perp = C^\perp \iff A = C$

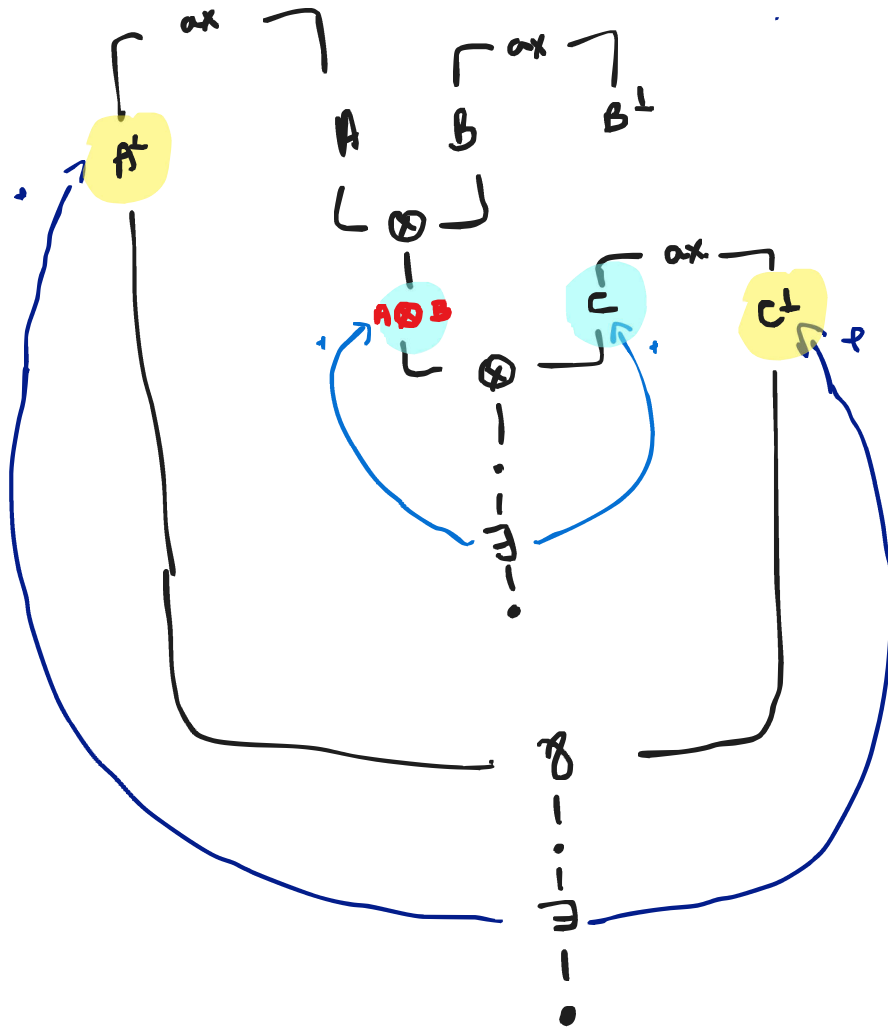
2- $C = A \otimes B$

$A = A \otimes B$

impossible!

$$\begin{array}{c}
 \frac{\frac{\overline{A, A^\perp} \quad \overline{B, B^\perp}}{A \otimes B, A^\perp, B^\perp} \otimes}{A \otimes B \otimes (A \otimes B), A^\perp, B^\perp, A^\perp \otimes B^\perp} \otimes \\
 \hline
 \exists z \, z \otimes z, A^\perp, B^\perp, A^\perp \otimes B^\perp \\
 \hline
 \exists z, z \otimes z, B^\perp, A^\perp \otimes (A^\perp \otimes B^\perp)
 \end{array}$$

Typing is non-trivial



CONSTRAINTS

1- $A^\perp = C^\perp \iff A = C$

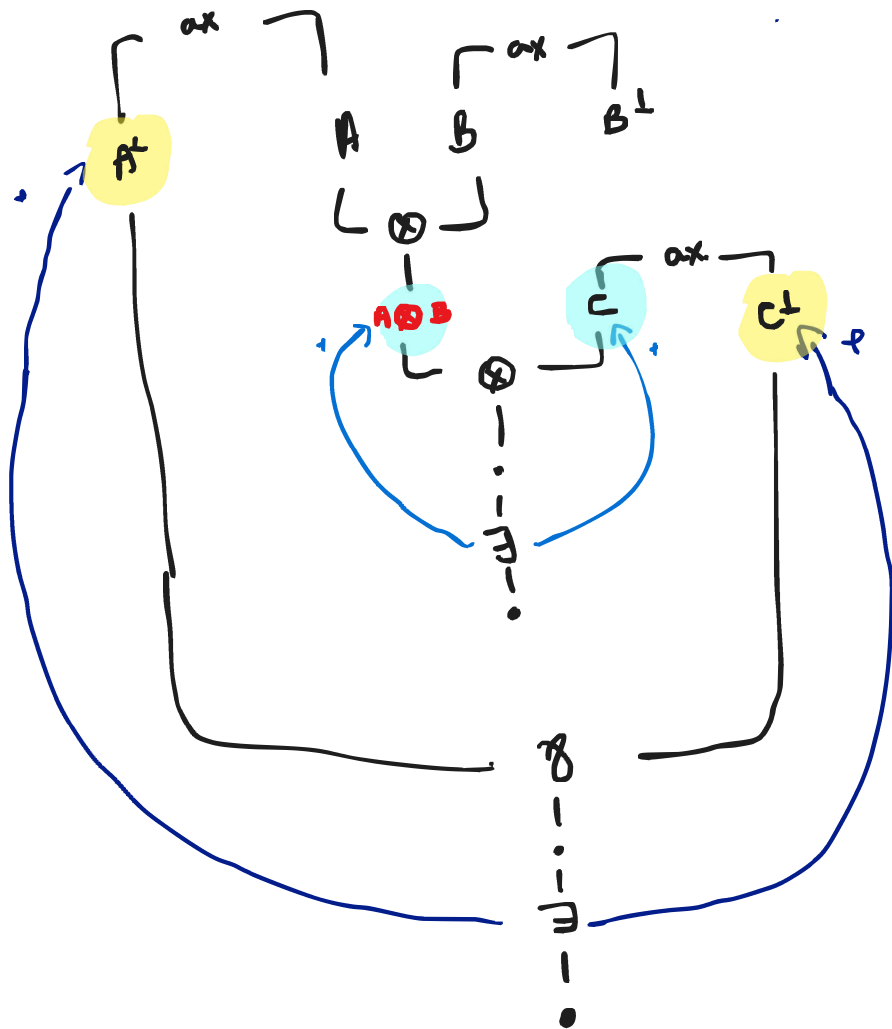
2- $C = A \otimes B$

$A = A \otimes B$

impossible!

$$\begin{array}{c}
 \frac{\frac{\overline{A, A^\perp} \quad \overline{B, B^\perp}}{A \otimes B, A^\perp, B^\perp} \otimes}{A \otimes B \otimes (A \otimes B), A^\perp, B^\perp, A^\perp \wp B^\perp} \otimes \\
 \frac{\exists z \, z \otimes z, A^\perp, B^\perp, A^\perp \wp B^\perp}{\exists z, z \otimes z, B^\perp, A^\perp \wp (A^\perp \wp B^\perp)} \exists \\
 \frac{\exists z, z \otimes z, B^\perp, \exists x \, x \wp x}{\exists z, z \otimes z, B^\perp, \exists x \, x \wp x} \exists \triangle
 \end{array}$$

Typing is non-trivial



CONSTRAINTS

4- $A^\perp = C^\perp \Leftrightarrow A = C$

2- $C = A \otimes B$

$$A = A \otimes B$$

impossible!

$$\begin{array}{l} \frac{A, A^\perp}{A \otimes B, A^\perp, B^\perp} \textcircled{1} \quad \frac{B, B^\perp}{A \otimes B, A^\perp \otimes B^\perp} \textcircled{2} \\ \hline A \otimes B \otimes (A \otimes B), A^\perp, B^\perp, A^\perp \otimes B^\perp \\ \hline \exists z \ z \otimes z, A^\perp, B^\perp, A^\perp \otimes B^\perp \\ \hline \exists z, z \otimes z, B^\perp, A^\perp \otimes (A^\perp \otimes B^\perp) \\ \hline \exists z, z \otimes z, B^\perp, \exists x \ x \otimes x \end{array}$$

3 A

TYPED VS UNTYPED

1 - Untyped nets (naive)

2 - How to type a net

3 - is atomic typing enough?

4 - typing is non-trivial

5 - Comparative

6 - Parsing as type inference

7 - More incorrectness

typed vs untyped: comparative

is a citation for atomically typed net enough?

typed vs untyped: comparative

is a criterion for atomically typed net enough?

typed net + axioms



η -expand

typed vs untyped: comparative

is a criterion for atomically typed net enough?

typed net + axioms



typed net + lemmas



η -expand

(proof search is unbounded)

typed vs untyped: comparative

is a criterion for atomically typed net enough?

typed net + axioms 

typed net + lemmas 

untyped net + axioms 

η -expand

(proof search is unbounded)

Find a generic type (not trivial)
+ η expand

typed vs untyped: comparative

is a criterion for atomically typed net enough?

typed net + axioms 

typed net + claimants 

untyped net + axioms 

untyped net + claimants 

η -expand

(proof search is unbounded)

Find a generic type (not trivial)
+ η expand

Find a generic type (not trivial)
+ proof search

typed vs untyped: comparative

is a criterion for atomically typed net enough?

typed net + axioms	✓	η -expand
typed net + claimants	✗ $\approx$	(proof search is unbounded)
untyped net + axioms	✓	Find a generic type (not trivial) + η expand
untyped net + claimants	✗ $\approx$	Find a generic type (not trivial) + proof search

typed vs untyped: comparative

is a criterion for atomically typed net enough?

typed net + axioms



η -expand

typed net + claimants



(proof search is unbounded)

untyped net + axioms



Find a generic type (not trivial)
+ η expand

untyped net + claimants



Find a generic type (not trivial)
+ proof search

typed vs untyped: comparative

is a criterion for atomically typed net enough?

typed net + axioms



typed net + claimants



untyped net + axioms



untyped net + claimants



η -expand

(proof search is unbounded)

Find a generic type (not trivial)
+ η expand

Find a generic type (not trivial)
+ proof search

typed vs untyped: comparative

is a criterion for atomically typed net enough?

untyped net + domains

$\approx$

Find a generic type (not trivial)
+ proof search

typed vs untyped: comparative

is a criterion for atomically typed net enough?

untyped net + domains

$\approx$

Find a generic type (not trivial)
+ proof search

↑
Criterion
here

typed vs untyped: comparative

is a criterion for atomically typed net enough?

untyped net + domains



Criterion
here

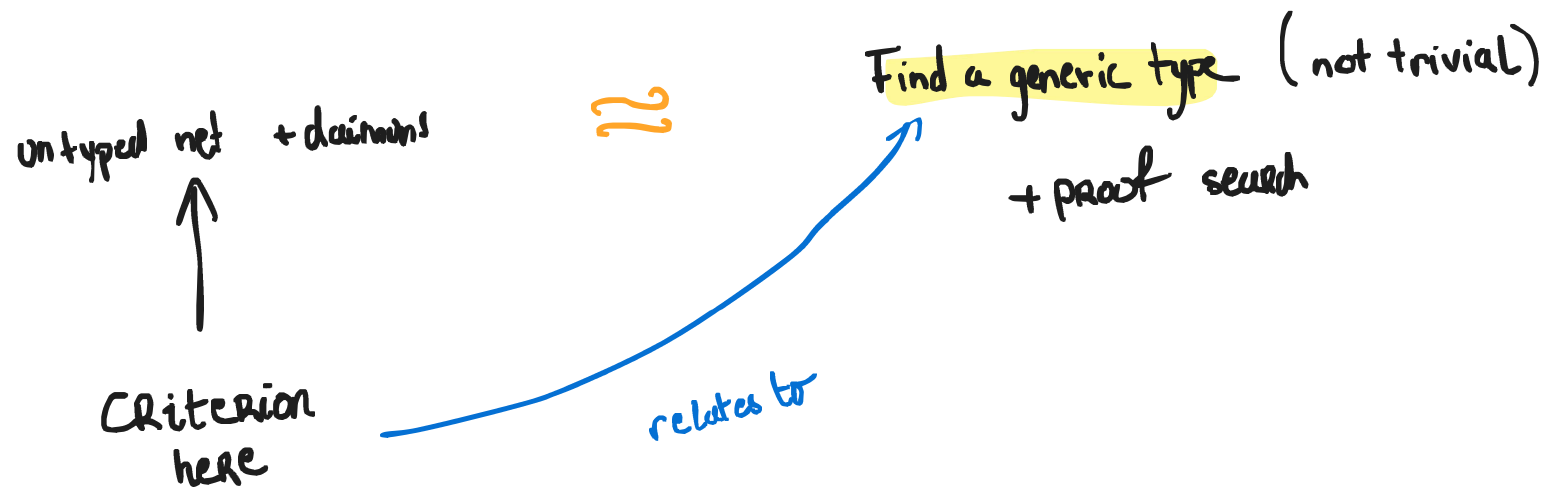
$\approx$

Find a generic type (not trivial)
+ proof search

relates to

typed vs untyped: comparative

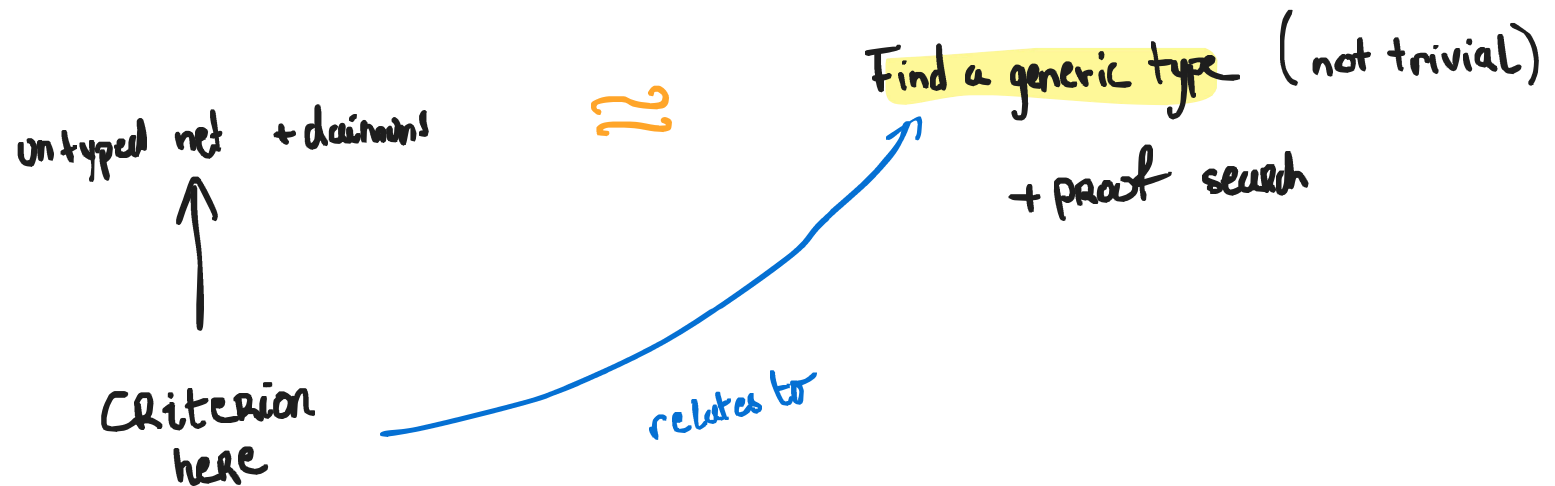
is a criterion for atomically typed net enough?



Deciding correctness with types is "enough"

typed vs untyped: comparative

is a criterion for atomically typed net enough?

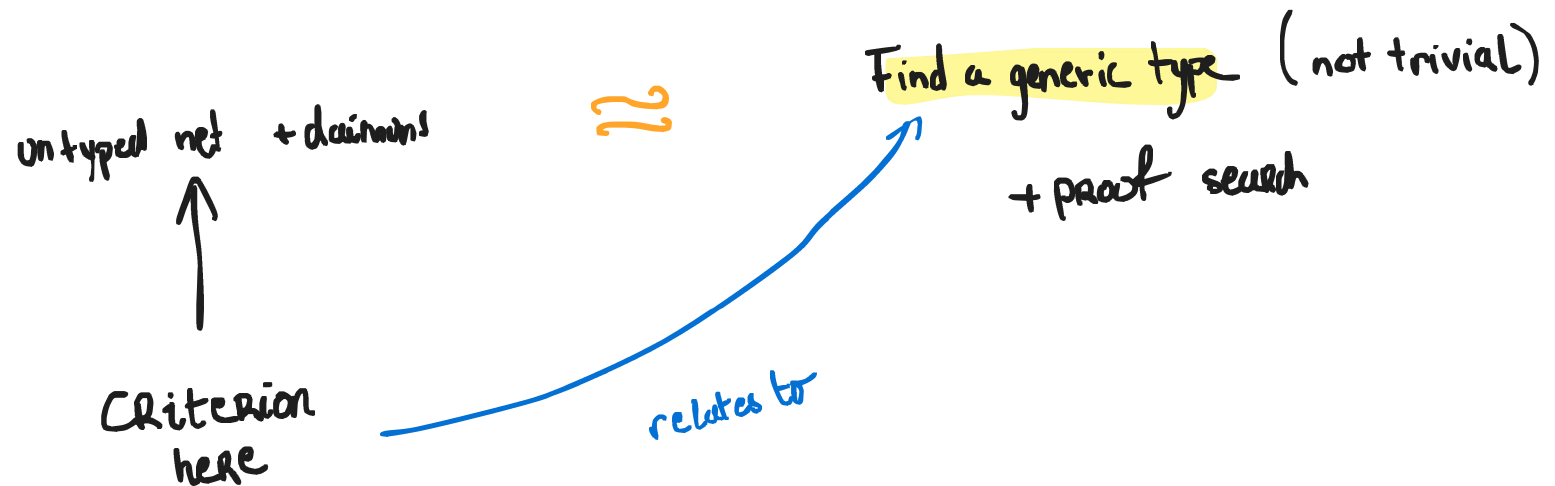


Deciding correctness with types is "enough"

↳ but types are not necessary to do so

typed vs untyped: comparative

is a criterion for atomically typed net enough?



Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

TYPED VS UNTYPED

- 1 - Untyped nets (naive)
- 2 - How to type a net
- 3 - is atomic typing enough?
- 4 - typing is non-trivial
- 5 - Comparative
- 6 - Parsing as type inference
- 7 - More incorrectness

Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

untyped λ -term t

Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

untyped λ -term t

(OCaml, Haskell)

Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

untyped λ -term t
(OCaml, Haskell)

syntactic analysis
→

Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

untyped λ -term t
(OCaml, Haskell)

syntactic analysis
→

$t:A$

+

$\triangle$
 $t:A$

Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

untyped λ -term t
(OCaml, Haskell)

syntactic analysis
→
(Hindley-Milner
algorithm)

$t:A$

+

$\triangleleft$
 $t:A$

Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

untyped λ -term t
(OCaml, Haskell)

syntactic analysis
→
(Hindley-Milner
algorithm)

$t:A$

+

$\triangleleft$
 $t:A$

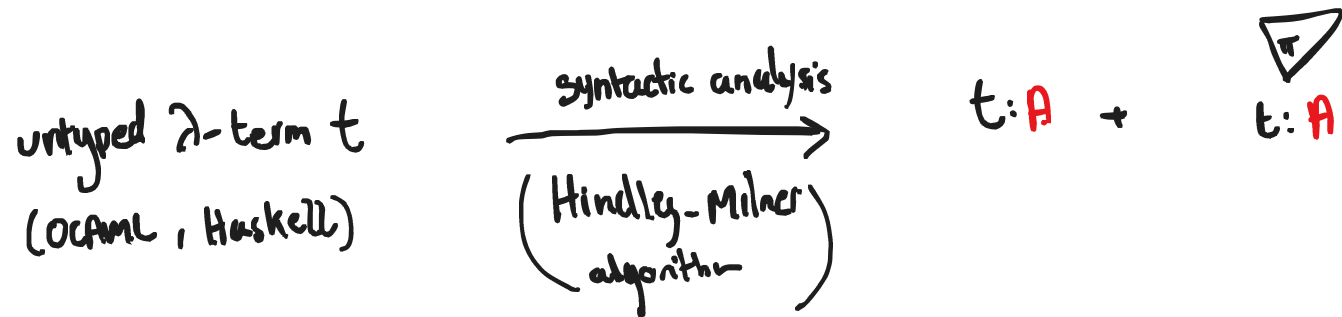
untyped net S

Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

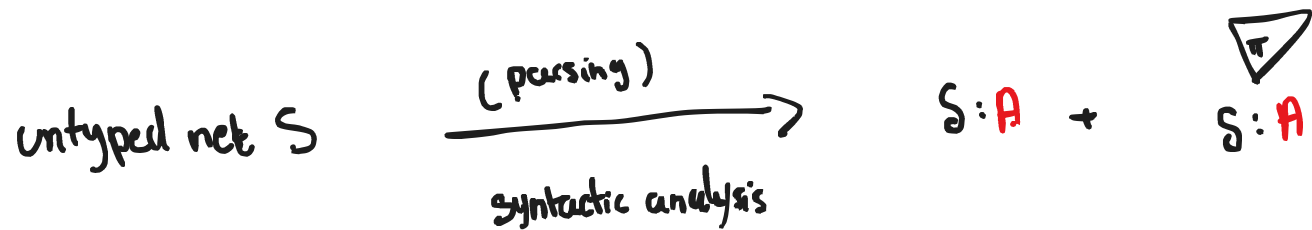
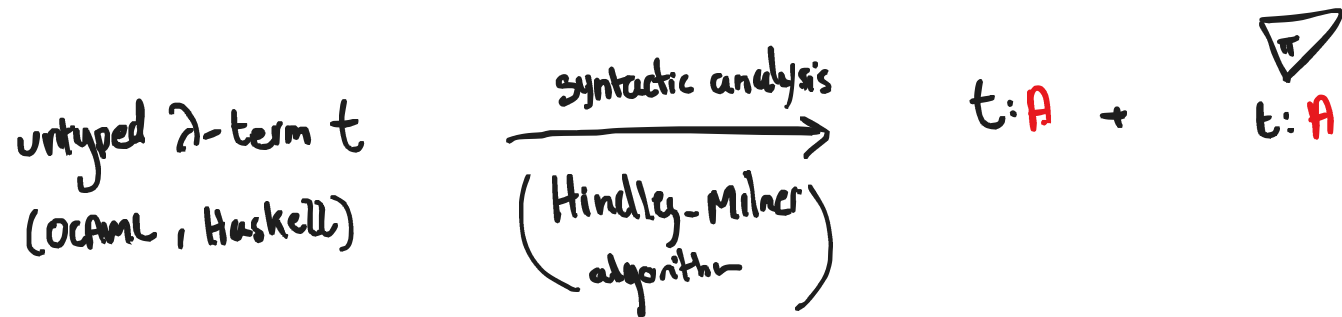


Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**



Parsing as type inference

Deciding correctness with types is "enough"

↳ but types are not necessary to do so

↳ w/o types deciding correctness relates to **type inference**

untyped λ -term t
(OCaml, Haskell)

OK

syntactic analysis
→
(Hindley-Milner
algorithm)

$t:A$ +

$\triangleleft$
 $t:A$

untyped net S

(parsing)
→
syntactic analysis

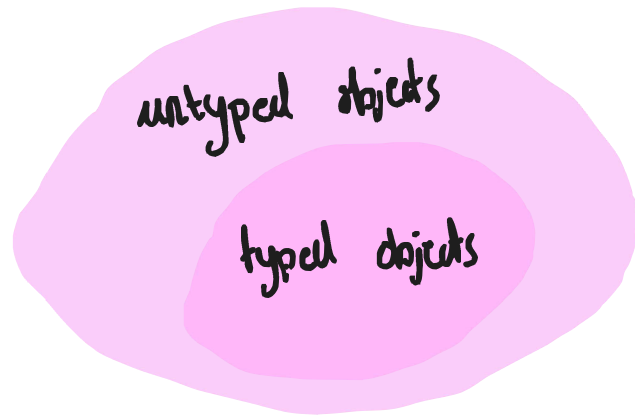
$S:A$ +

$\triangleleft$
 $S:A$

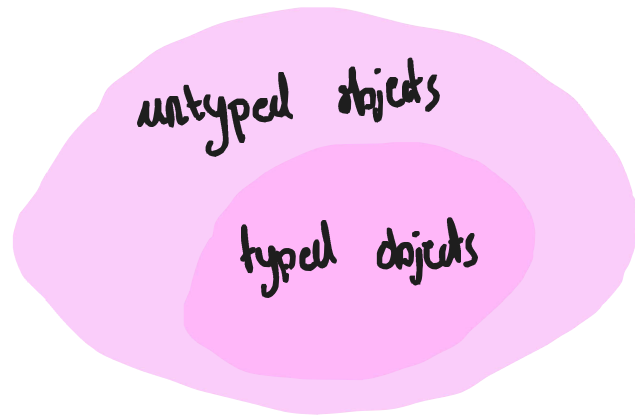
TYPED VS UNTYPED

- 1 - Untyped nets (naive)
- 2 - How to type a net
- 3 - is atomic typing enough?
- 4 - typing is non-trivial
- 5 - Comparative
- 6 - Parsing as type inference
- 7 - More incorrectness

More incorrectness

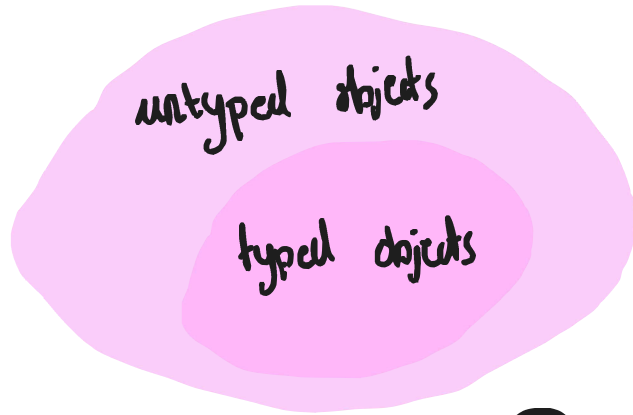


More incorrectness



in λ :

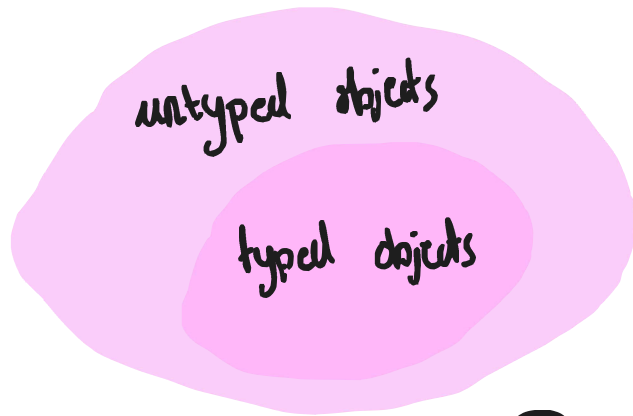
More incorrectness



in λ :

$\lambda x \text{ (}\lambda x$

More incorrectness

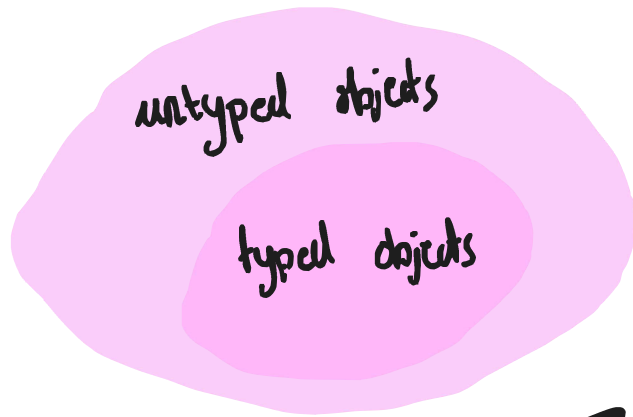


in λ :

$\lambda x (\lambda x \leftarrow$

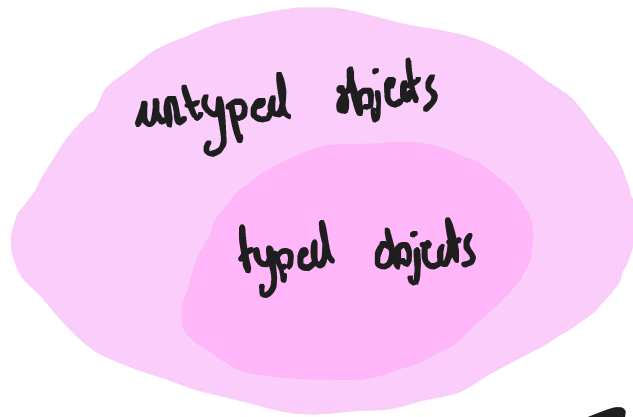
not typeable 'à la Curry'

More incorrectness



For proof nets as well!

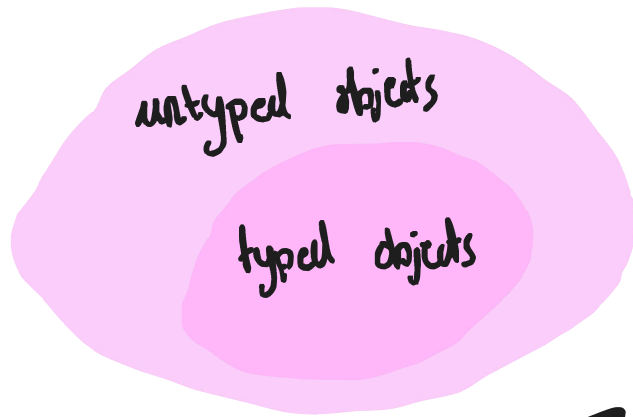
More incorrectness



For proof nets as well!

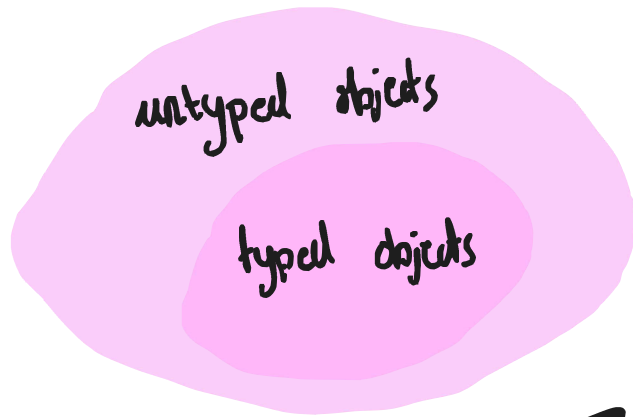
— cuts

More incorrectness



For proof nets as well!
– cuts (cycle)

More incorrectness

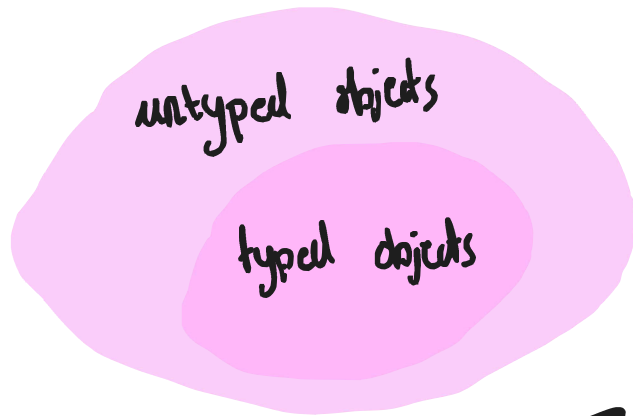


For proof nets as well!

- cuts (cycle)

- $\exists$

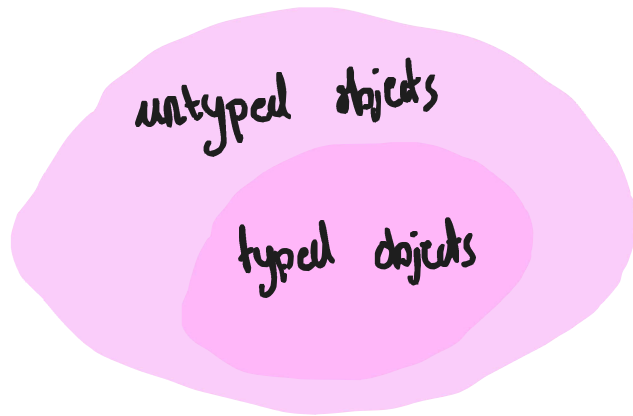
More incorrectness



For proof nets as well!

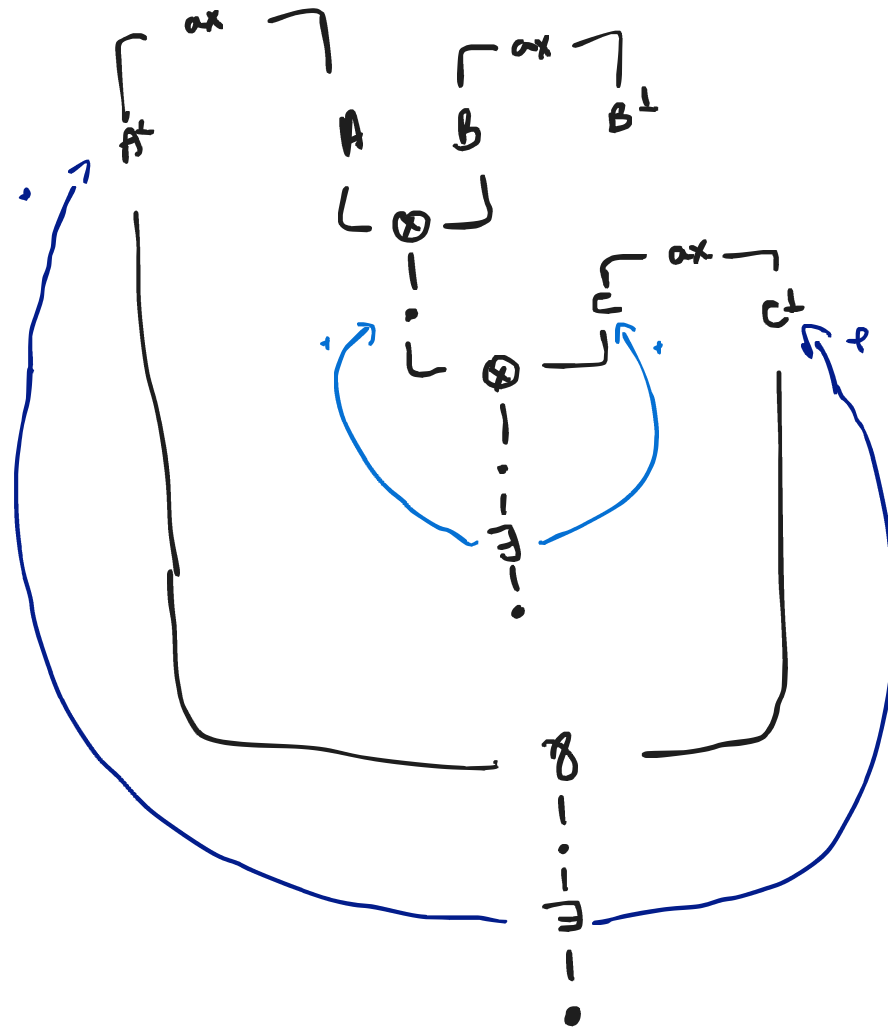
- cuts (cycle)
- $\exists$ (no cycle)

More incorrectness



For proof nets as well!

- cuts (cycle)
- $\exists$ (no cycle)



incorrect
+
untypeable
+
cut free

PLAN

I MLL2 Nets : challenges

II Typed vs untyped nets

- OUR CRITERION

III Nets : desequentializing proofs

- OUR CRITERION

IV PARSING strategy

V PARSING (naive)

VI PARSING with schemes

PLAN

I MLL2 Nets : challenges

II Typed vs untyped nets

- OUR CRITERION 

III Nets : desequentializing proofs

- OUR CRITERION

IV PARSING strategy

V PARSING (naive)

VI PARSING with schemes

MAIN RESULT

$$S = [\pi]$$

↑ proof of ML2

$\Leftrightarrow$

$$S \xrightarrow[p]{*} \begin{bmatrix} * \\ 1 \dots n \end{bmatrix}$$

MAIN RESULT

$$S = [\pi]$$

↑ proof of $ML2^\pm$

$\Leftrightarrow$

$$S \xrightarrow[p]{*} \begin{bmatrix} * \\ 1 \dots n \end{bmatrix}$$

MAIN RESULT

SOUNDNESS

$$S = [\pi]$$

↑ proof of ML2

$\Rightarrow$

$$S \xrightarrow[p]{*} \begin{bmatrix} \vdots \\ 1 \dots n \end{bmatrix}$$

MAIN RESULT

SOUNDNESS

$$S = [\pi] \quad \Rightarrow \quad S \xrightarrow[p]{*} \begin{bmatrix} * \\ 1 \dots n \end{bmatrix}$$

$\uparrow$ proof of ML2

COMPLETENESS

$$S \xrightarrow[p]{*} \begin{bmatrix} * \\ 1 \dots n \end{bmatrix} \quad \Rightarrow \quad S = [\pi]$$

$\uparrow$ proof of ML2

MAIN RESULT

define its class

SOUNDNESS

$$S = [\pi] \quad \Rightarrow \quad S \xrightarrow[p]{*} \begin{bmatrix} \vdots \\ 1 \dots n \end{bmatrix}$$

$\uparrow$ proof of ML2

COMPLETENESS

$$S \xrightarrow[p]{*} \begin{bmatrix} \vdots \\ 1 \dots n \end{bmatrix} \quad \Rightarrow \quad S = [\pi]$$

$\uparrow$ proof of ML2

MAIN RESULT

SOUNDNESS

define its class

define $\llbracket \cdot \rrbracket$

$$S = \llbracket \pi \rrbracket$$

↑ proof of ML2

$\Rightarrow$

$$S \rightarrow_p^* \begin{bmatrix} \tau \\ 1 \dots n \end{bmatrix}$$

COMPLETENESS

$$S \rightarrow_p^* \begin{bmatrix} \tau \\ 1 \dots n \end{bmatrix}$$

$\Rightarrow$

$$S = \llbracket \pi \rrbracket$$

↑ proof of ML2

MAIN RESULT

SOUNDNESS

define its class

define $\llbracket \cdot \rrbracket$

define the $\rightarrow_p^*$ reduction

$$S = \llbracket \pi \rrbracket$$

↑ proof of ML2

$\Rightarrow$

$$S \rightarrow_p^* \begin{bmatrix} \tau \\ 1 \dots n \end{bmatrix}$$

COMPLETENESS

$$S \rightarrow_p^* \begin{bmatrix} \tau \\ 1 \dots n \end{bmatrix}$$

$\Rightarrow$

$$S = \llbracket \pi \rrbracket$$

↑ proof of ML2

MAIN RESULT

SOUNDNESS

define its class

$$S = [\pi]$$

↑ proof of ML2

define $\llbracket \cdot \rrbracket$

$\Rightarrow$

$$S \rightarrow_p^* \begin{bmatrix} \tau \\ 1 \dots n \end{bmatrix}$$

define the $\rightarrow_p$ reduction

COMPLETENESS

$$S \rightarrow_p^* \begin{bmatrix} \tau \\ 1 \dots n \end{bmatrix}$$

$\Rightarrow$

$$S = [\pi]$$

(Constructive)

↑ proof of ML2

PLAN

I MLL2 Nets : challenges

II Typed vs untyped nets

- OUR CRITERION

III Nets : desequentializing proofs



- OUR CRITERION

IV PARSING strategy

V PARSING (naive)

VI PARSING with schemes

MLL₂⁺ Nets

MU2+ Pae-nets

pae-Net

MU2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, e)

MU2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, e)

position



MU2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, e)



position links

MU2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, e)

position $\nearrow$ LINKS $\uparrow$ $\uparrow$ $s: E \rightarrow P_s^{\text{fin}}(V)$ (SOURCE)

MU2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (target)

$s: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (source)

position $\nearrow$

links $\uparrow$

MU2+ Pae-nets

pae-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (source)

position $\uparrow$ links

MU2+ Pae-nets

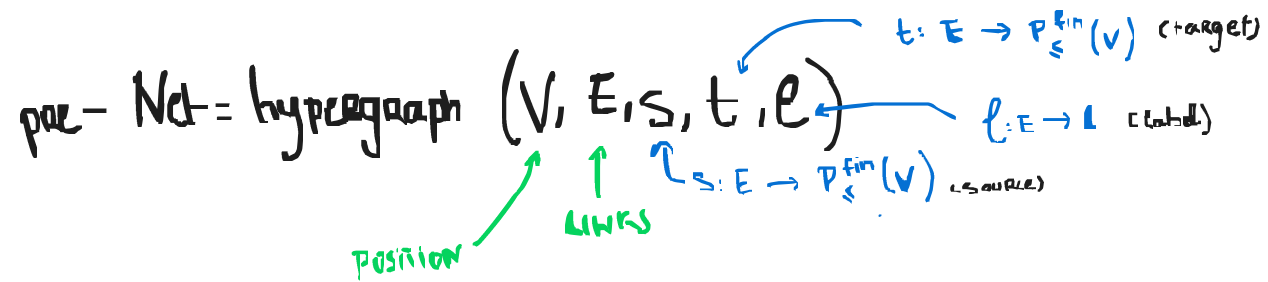
pae-Net = hypergraph (V, E, s, t, ℓ)

$t: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (source)

position $\uparrow$ links

links:

MU2+ Pae-nets



links:

$$\begin{matrix} 1 & \dots & n \\ \uparrow & & \uparrow \\ 1 & \dots & n \end{matrix}$$

MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\subseteq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\begin{matrix} \lceil & + & \rceil \\ 1 & \dots & n \end{matrix}$

$\begin{matrix} p_1 & & p_2 \\ \downarrow & \downarrow & \downarrow \\ L & \emptyset & J \\ & \downarrow & \\ & P & \end{matrix}$

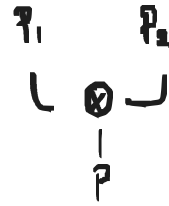
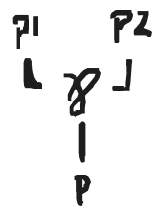
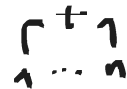
MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:



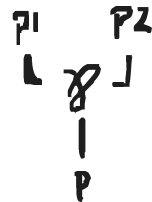
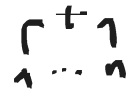
MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:



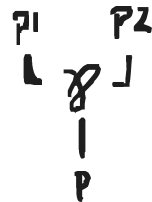
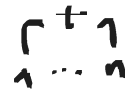
MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:



MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

Links:

$\begin{matrix} \uparrow \\ 1 \dots n \end{matrix}$

$\begin{matrix} p_1 & p_2 \\ \downarrow & \downarrow \\ \emptyset & \downarrow \\ \downarrow & \downarrow \\ p & \end{matrix}$

$\begin{matrix} p_1 & p_2 \\ \downarrow & \downarrow \\ \emptyset & \downarrow \\ \downarrow & \downarrow \\ p & \end{matrix}$

$\begin{matrix} p \\ \downarrow \\ \vee \\ \downarrow \\ q \end{matrix}$

$\begin{matrix} p \\ \downarrow \\ \exists \\ \downarrow \\ q \end{matrix}$

$\begin{matrix} p_1 & p_2 \\ \downarrow & \downarrow \\ \text{cut} & \end{matrix}$

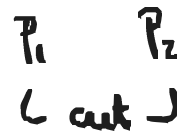
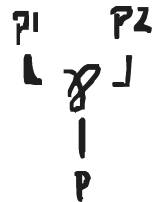
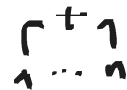
MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

links:



examples

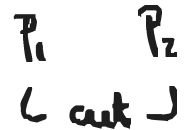
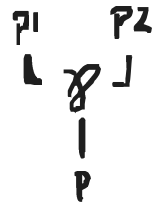
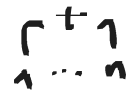
MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

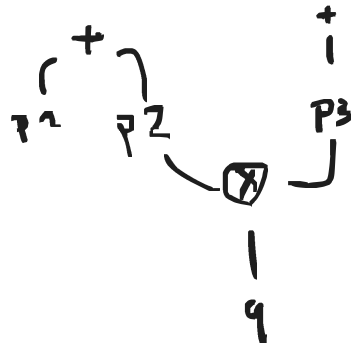
$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

links:



examples



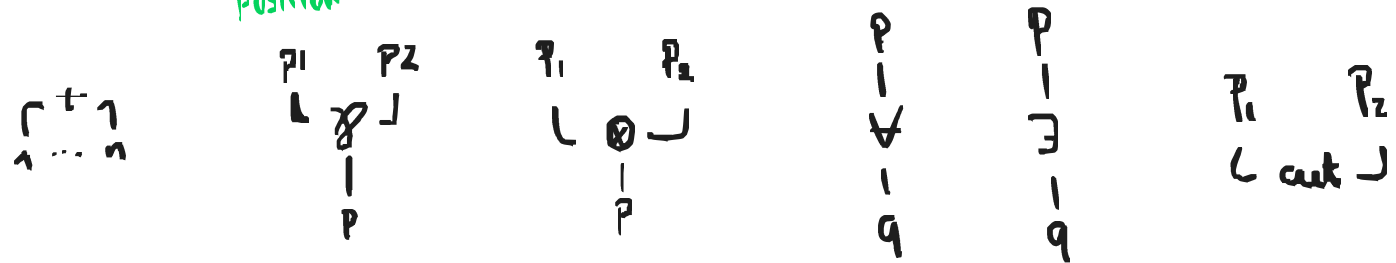
MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

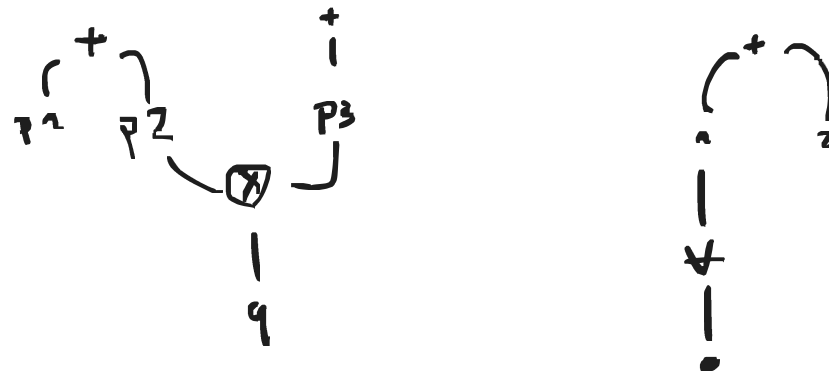
$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

links:



examples



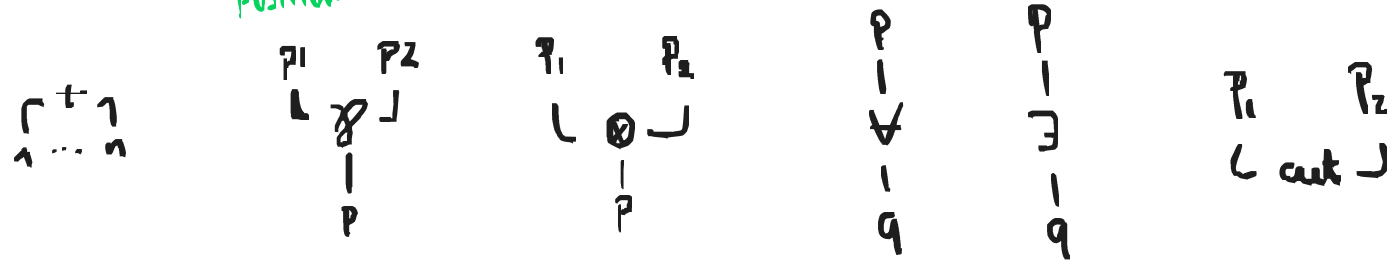
MU2+ Pae-nets

$\text{pae-Net} = \text{hypergraph } (V, E, s, t, \ell)$

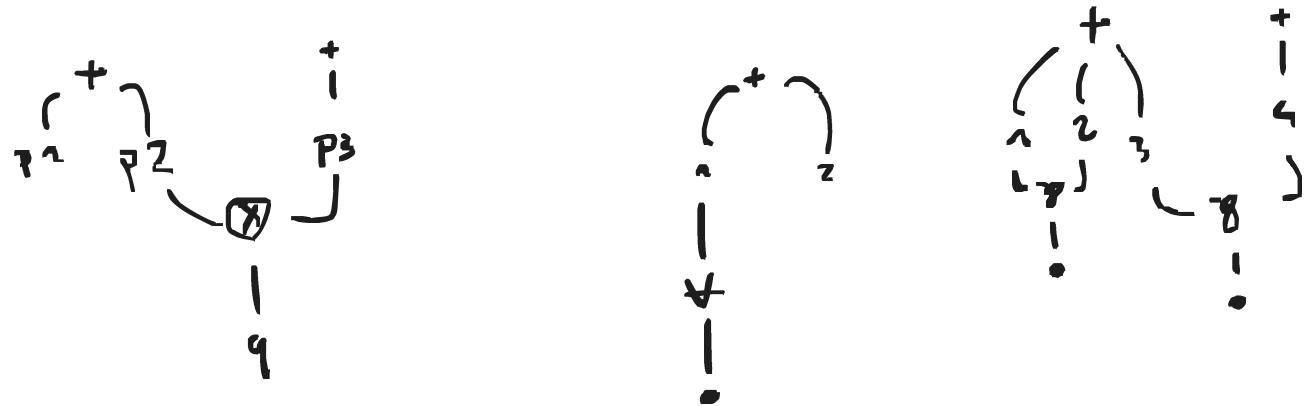
$t: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (target)
 $\ell: E \rightarrow L$ (label)
 $s: E \rightarrow P_{\leq}^{\text{fin}}(V)$ (source)

position $\uparrow$
 links $\uparrow$

links:



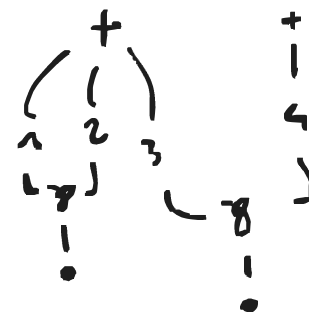
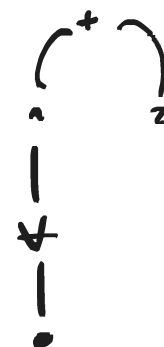
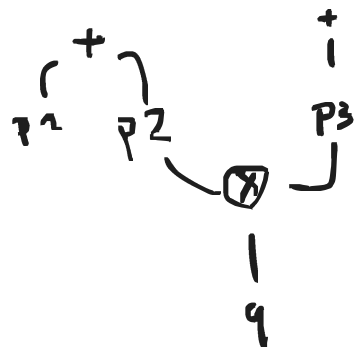
examples



ML2+ Nets

Net =

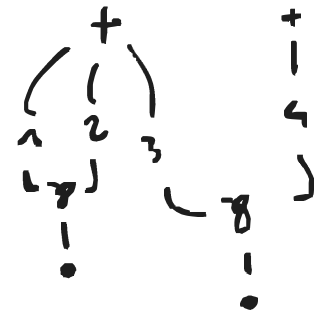
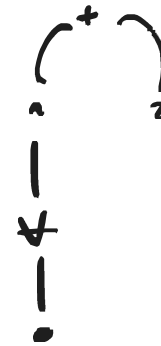
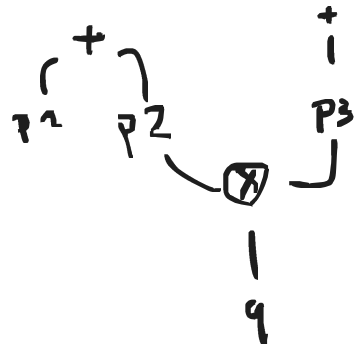
examples
of pre-nets



ML2+ Nets

Net = pre-net

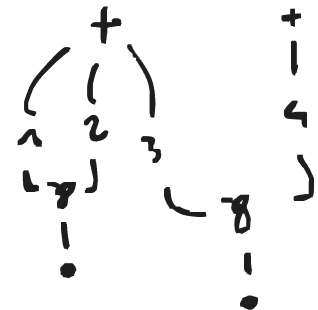
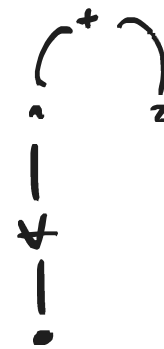
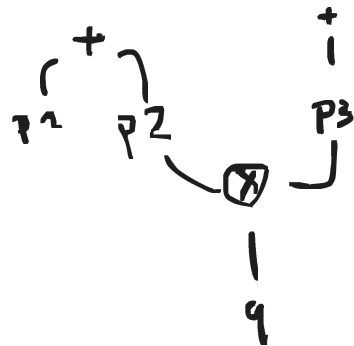
examples
of pre-nets



MU2+ Nets

Net = pre-net + pointers

examples
of pre-nets

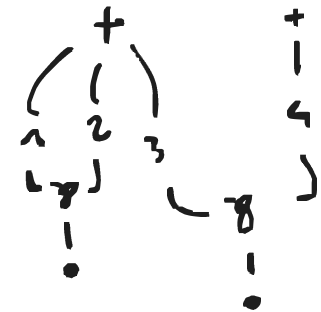
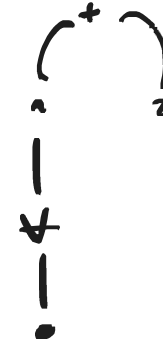
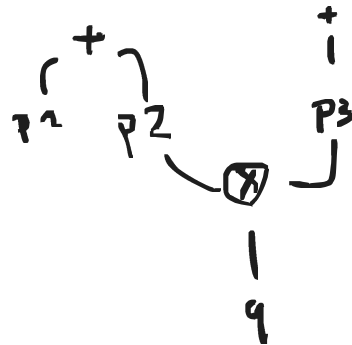


MU2+ Nets

Net = pre-net + pointers ↑
tuple

(e, x, λ, s)
V or $\exists$ link $\rightarrow e$
position or $\exists$ link. $\rightarrow x$
status 0 or 1. $\rightarrow s$
polarity + or - $\rightarrow \lambda$

examples
of pre-nets

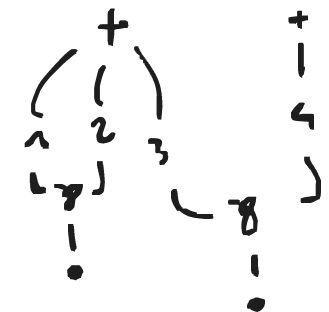
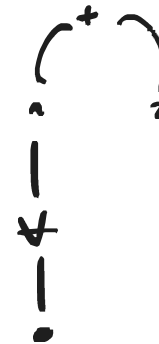
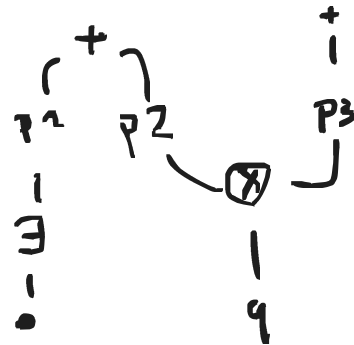


MU2+ Nets

Net = pre-net + pointers ↑
tuple

(e, x, λ, s)
V or $\exists$ link $\rightarrow e$
position or $\exists$ link. $\rightarrow x$
status 0 or 1. $\rightarrow \lambda$
polarity + or - $\rightarrow s$

examples
of pre-nets

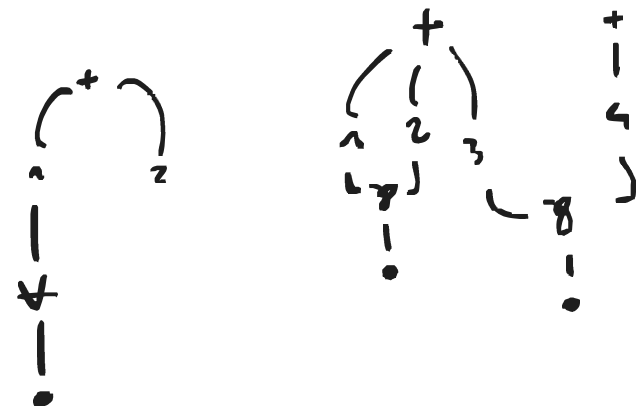
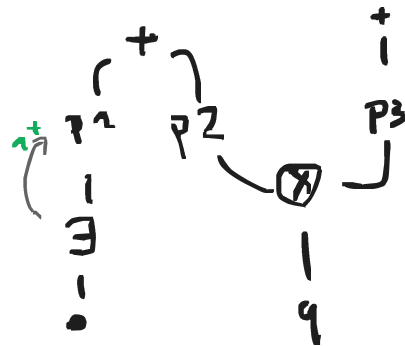


MU2+ Nets

Net = pre-net + pointers ↑
tuple

(e, x, λ, s)
V or $\exists$ link $\rightarrow e$
position or $\exists$ link. $\rightarrow x$
status 0 or 1. $\rightarrow s$
polarity + or - $\rightarrow \lambda$

examples
of ~~pre~~-nets

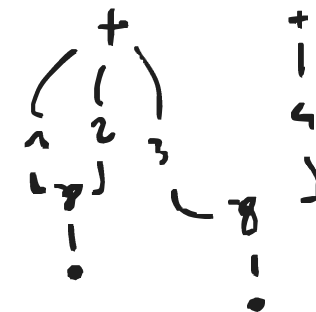
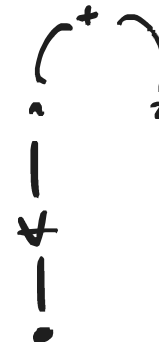
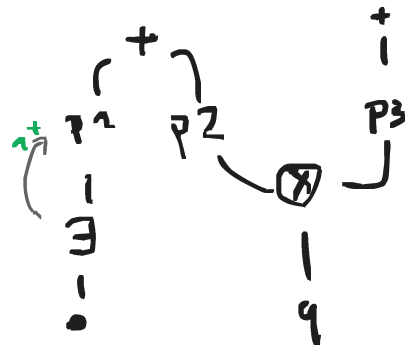


MU2+ Nets

Net = pre-net + pointers ↑ tuple

(e, x, λ, s)
V or $\exists$ link $\rightarrow e$
position or $\exists$ link. $\rightarrow x$
status 0 or 1. $\rightarrow \lambda$
polarity + or - $\rightarrow s$

examples
of ~~pre~~-nets

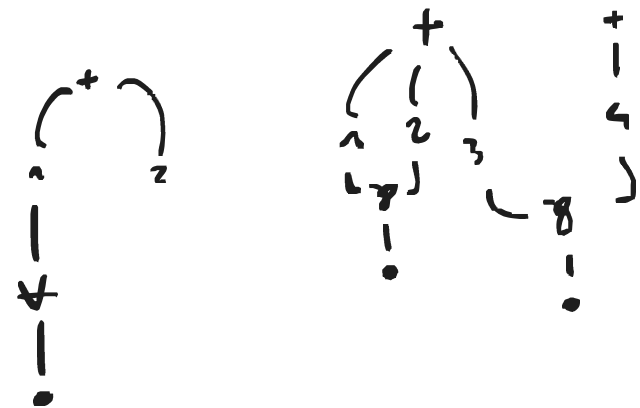
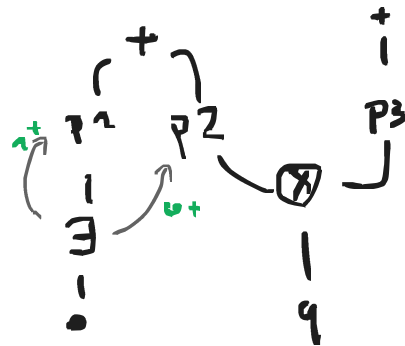


MU2+ Nets

Net = pre-net + pointers ↑ tuple

(e, x, λ, s)
V or $\exists$ link $\rightarrow e$
position or $\exists$ link. $\rightarrow x$
status 0 or 1. $\rightarrow \lambda$
polarity + or - $\rightarrow s$

examples
of ~~pre~~-nets

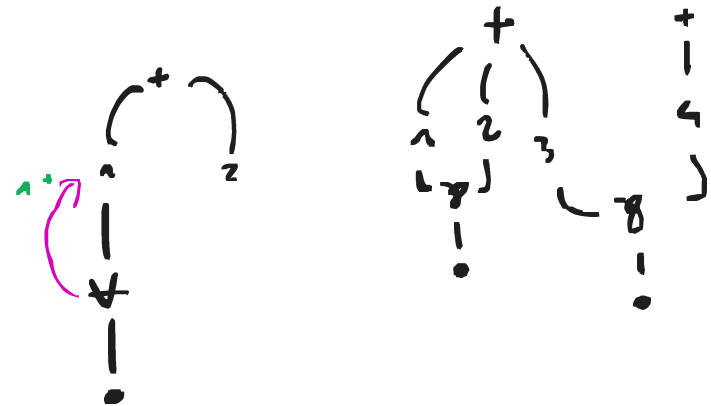
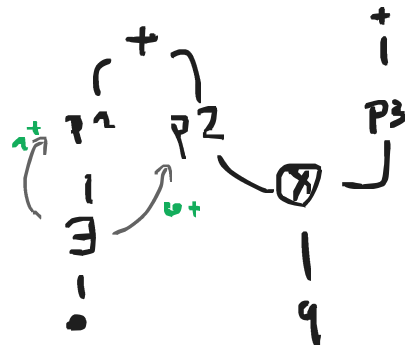


MU2+ Nets

Net = pre-net + pointers ↑ tuple

(e, x, λ, s)
V or $\exists$ link $\rightarrow e$
position or $\exists$ link. $\rightarrow x$
status 0 or 1. $\rightarrow \lambda$
polarity + or - $\rightarrow s$

examples
of ~~pre~~-nets

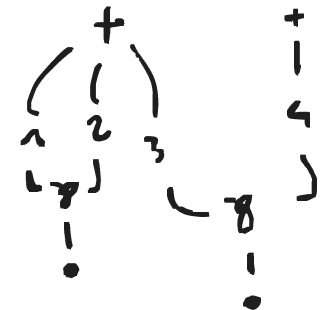
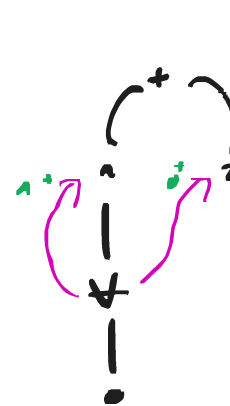
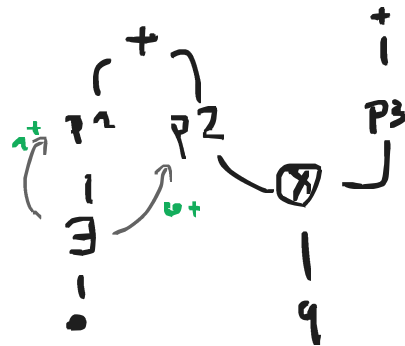


MU2+ Nets

Net = pre-net + pointers ↑
tuple

(e, x, λ, s)
↑ V or $\exists$ link
↑ position or $\exists$ link.
↑ status 0 or 1.
↑ polarity + or -

examples
of ~~pre~~-nets

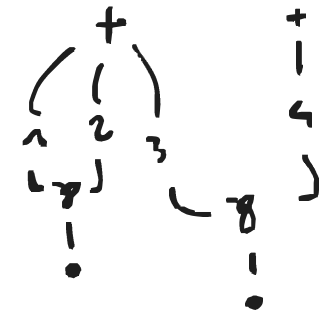
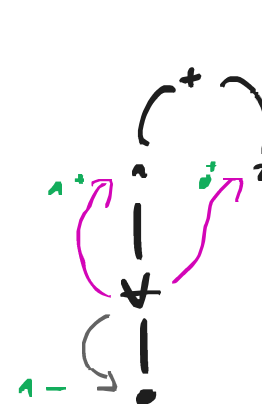
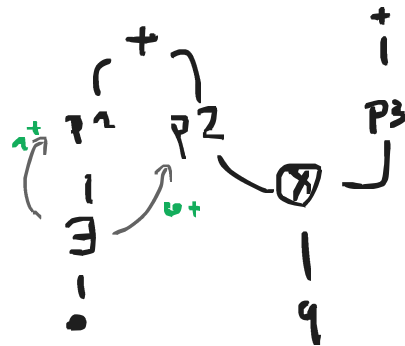


MU2+ Nets

Net = pre-net + pointers ↑ tuple

(e, x, λ, s)
↑ $\forall$ or $\exists$ link
↑ position or $\exists$ link.
↑ status 0 or 1.
↑ polarity + or -

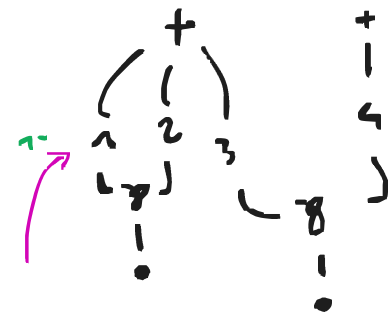
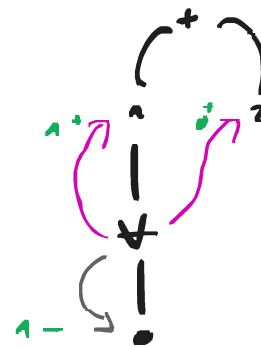
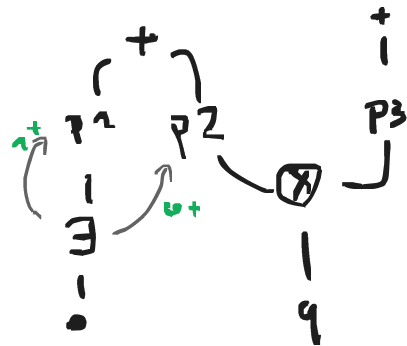
examples
of ~~pre~~-nets



MLL2+ Nets

Net = pre-net + pointers

examples
of ~~pre~~-nets

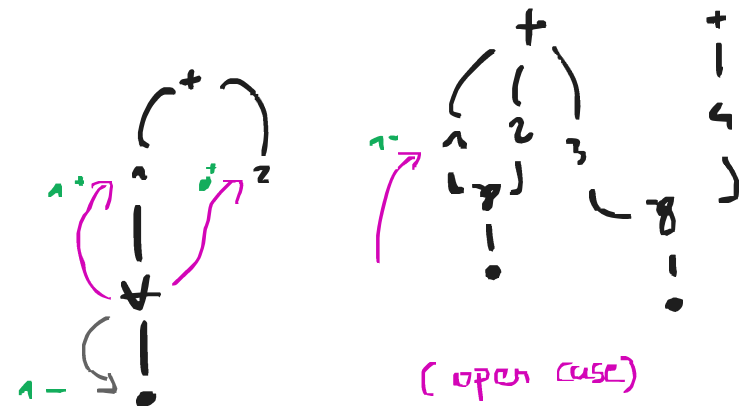
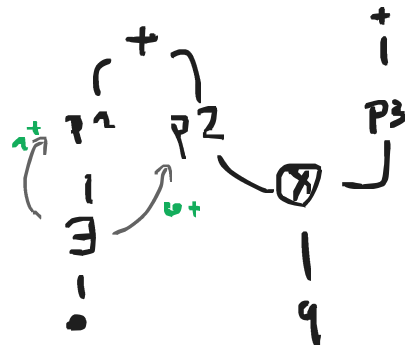


MU2+ Nets

Net = pre-net + pointers ↑
tuple

(e, x, λ, s)
↑ $\forall$ or $\exists$ link
↑ position or $\exists$ link.
↑ status 0 or 1.
↑ polarity + or -

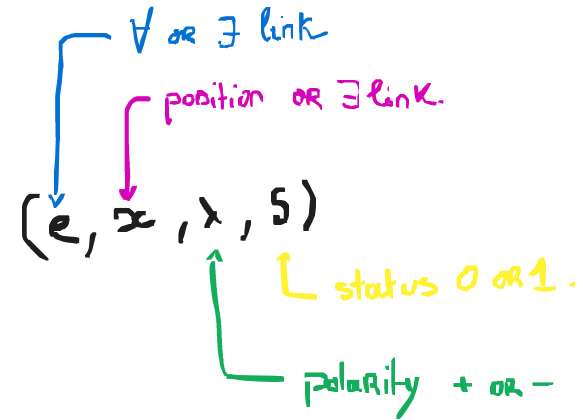
examples
of ~~pre~~-nets



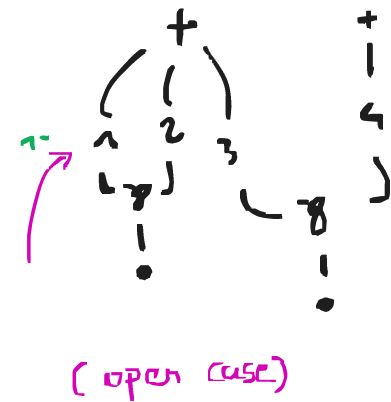
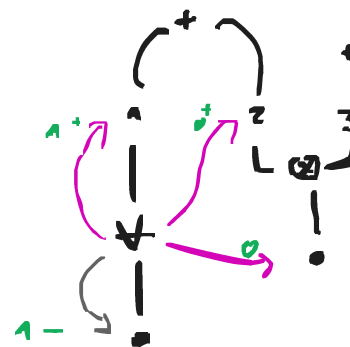
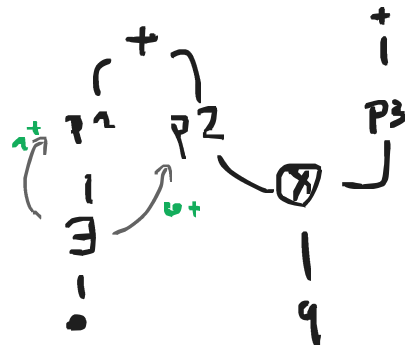
MU2+ Nets

Net = pre-net + pointers

tuple



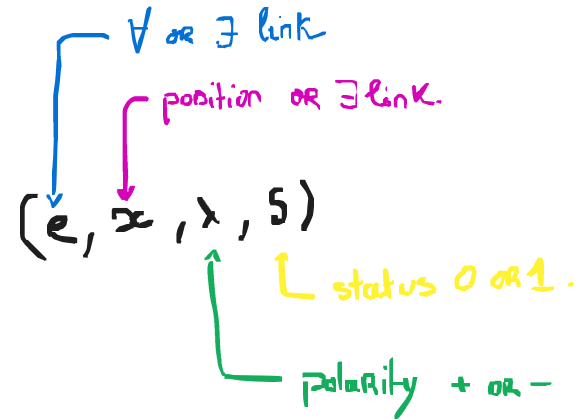
examples
of pre-nets



MU2+ Nets

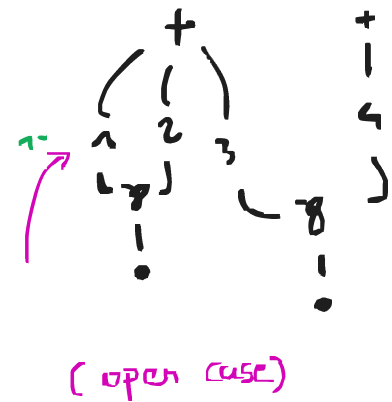
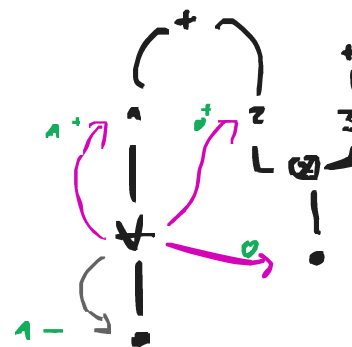
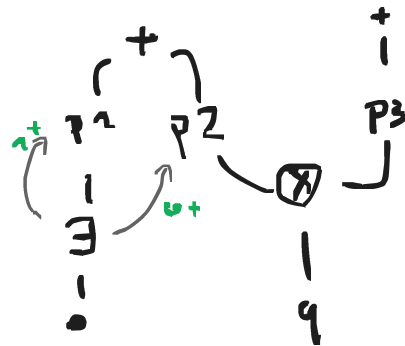
Net = pre-net + pointers

tuple



pointer restriction:

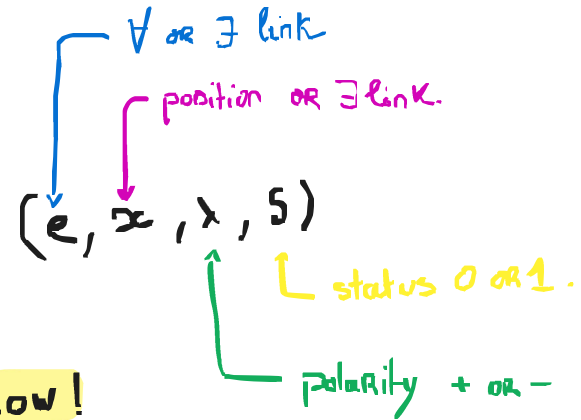
examples
of ~~pre~~-nets



MU2+ Nets

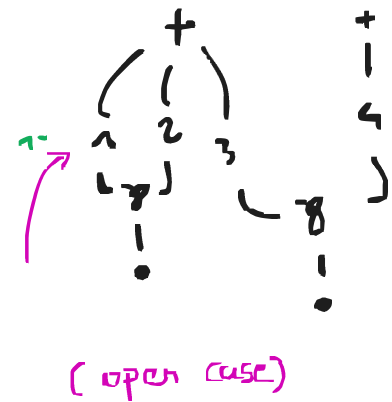
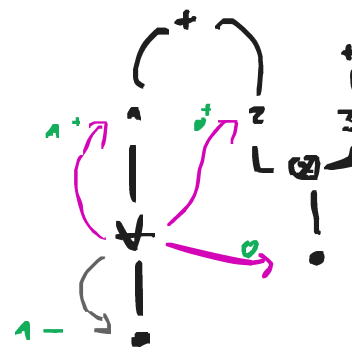
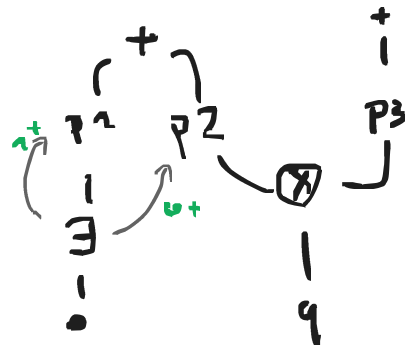
Net = pre-net + pointers

tuple



pointer restriction: ① **DON'T POINT BELOW!**

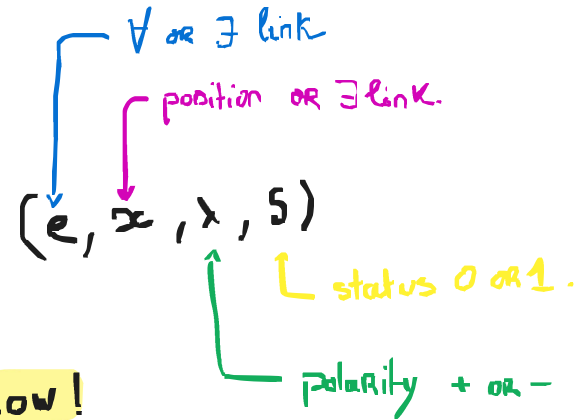
examples
of pre-nets



MU2+ Nets

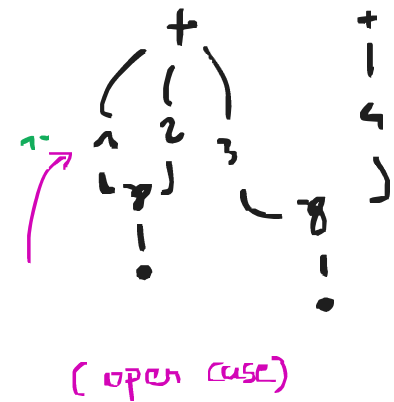
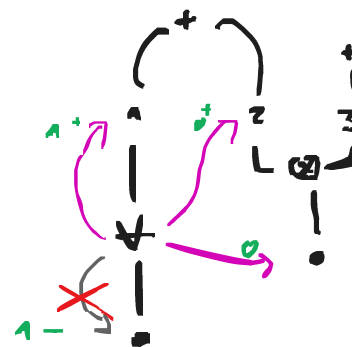
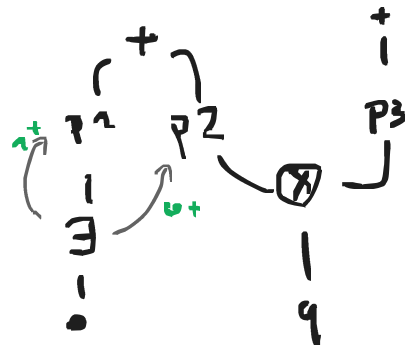
Net = pre-net + pointers

tuple



pointer restriction: ① **DON'T POINT BELOW!**

examples
of ~~pre~~-nets

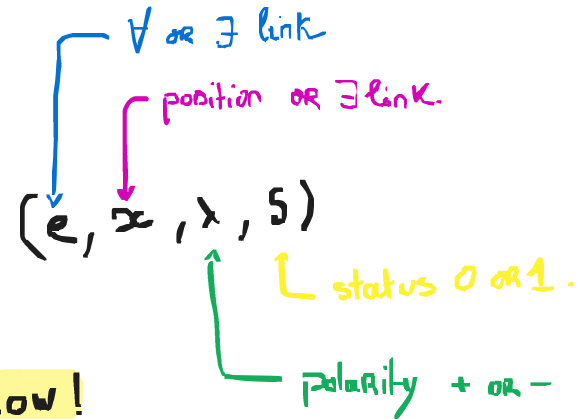


(open case)

MU2+ Nets

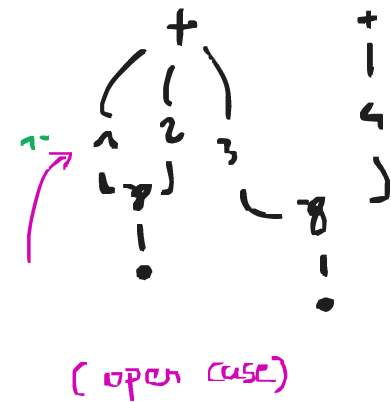
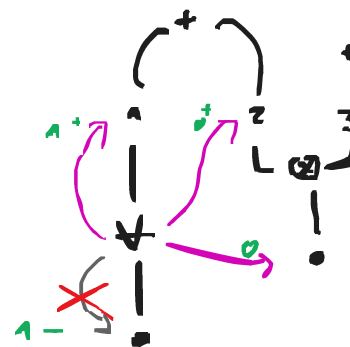
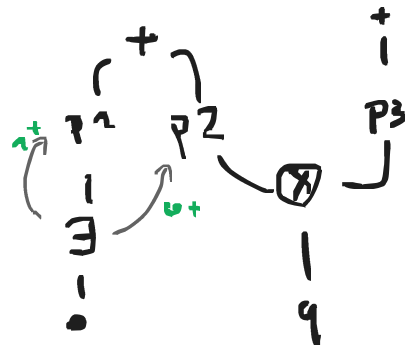
Net = pre-net + pointers

tuple



- pointer Restriction:
- ① DON'T POINT BELOW!
 - ② $\exists$ -pointers ONLY ABOVE!

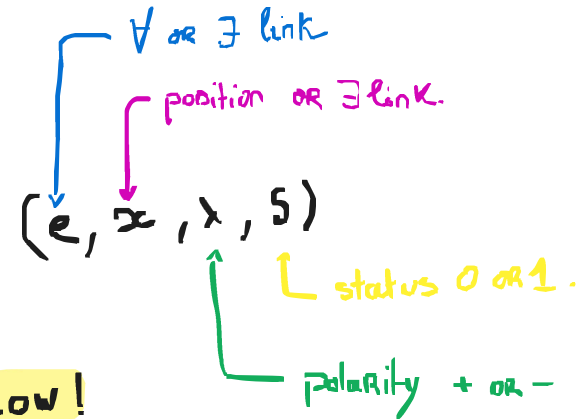
examples
of pre-nets



MU2+ Nets

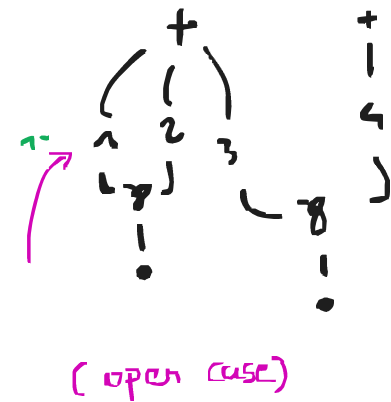
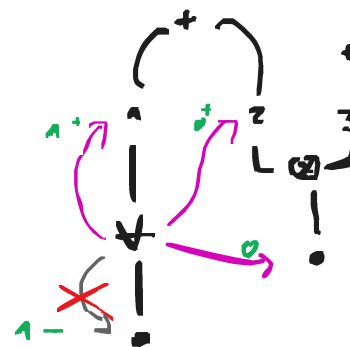
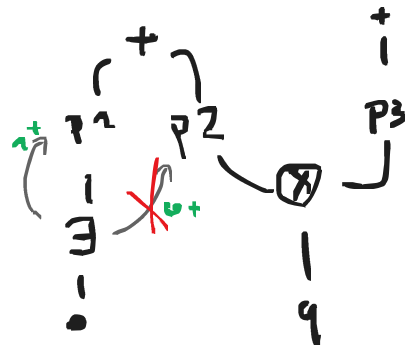
Net = pre-net + pointers

tuple



- pointer Restriction:
- ① DON'T POINT BELOW!
 - ② $\exists$ -pointers ONLY ABOVE!

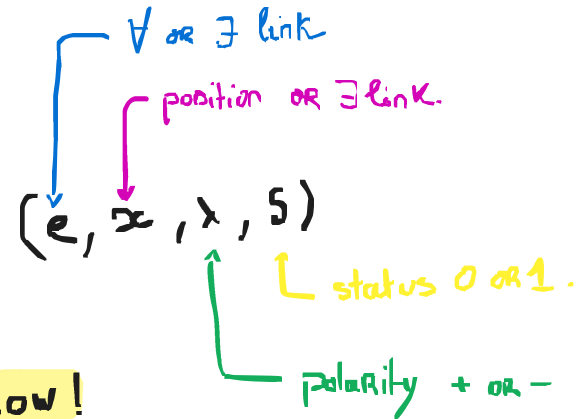
examples
of ~~pre~~-nets



MU2+ Nets

Net = pre-net + pointers

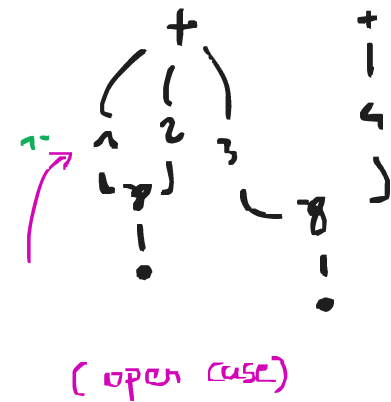
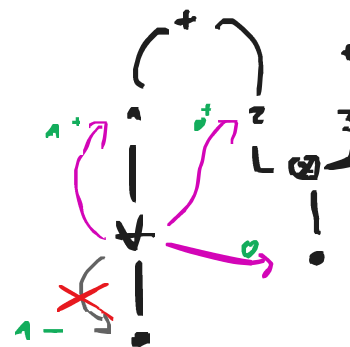
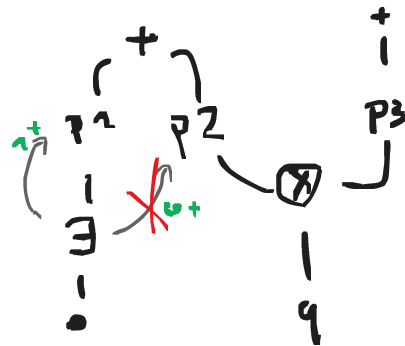
tuple



pointer Restriction:

- ① DON'T POINT BELOW!
- ② $\exists$ -pointers ONLY ABOVE!
- ③ $\forall$ -pointer ONLY ON $\forall$ -OUTPUT / $\exists$ -link

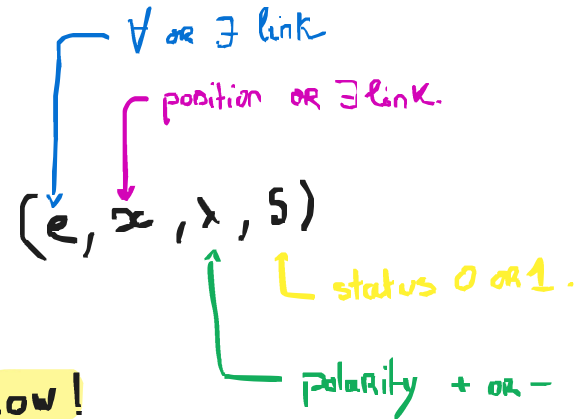
examples
of pre-nets



MU2+ Nets

Net = pre-net + pointers

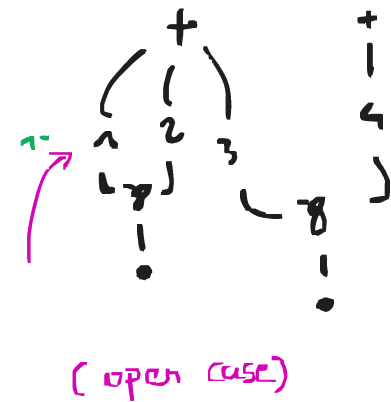
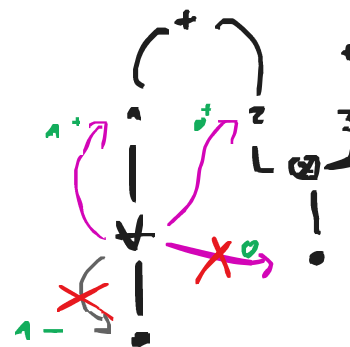
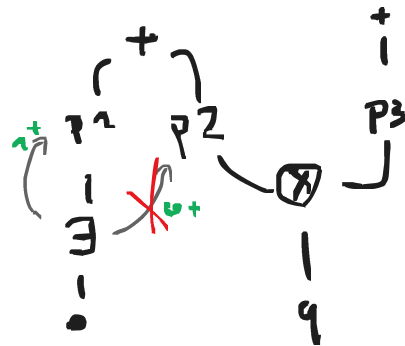
tuple



pointer Restriction:

- ① DON'T POINT BELOW!
- ② $\exists$ -pointers ONLY ABOVE!
- ③ $\forall$ -pointer ONLY ON $\forall$ -OUTPUT / $\exists$ -link

examples
of ~~pre~~-nets



Coherence of Witnesses

Coherence of Witnesses

Coherence of Witnesses

$T_1 \cap T_2$ iff $\exists s$: T_1 and T_2 typeable by A and

Coherence of Witnesses

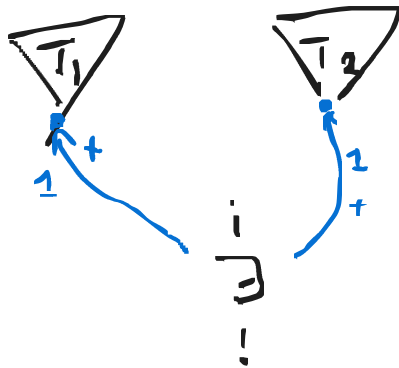
$T_1 \cap T_2$ iff $\exists S$: T_1 and T_2 typeable by A and

$T_1 \cup T_2$ iff $\exists S$: T_1 typ. A and T_2 typ. $A^\perp$ and

Coherence of Witnesses

$T_1 \cap T_2$ iff $\exists S$: T_1 and T_2 typeable by A and

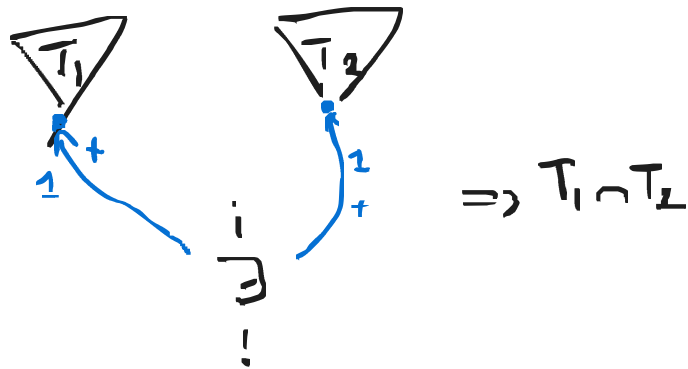
$T_1 \cup T_2$ iff $\exists S$: T_1 typ. A and T_2 typ. $A^\perp$ and



Coherence of Witnesses

$T_1 \cap T_2$ iff $\exists S$: T_1 and T_2 typeable by A and

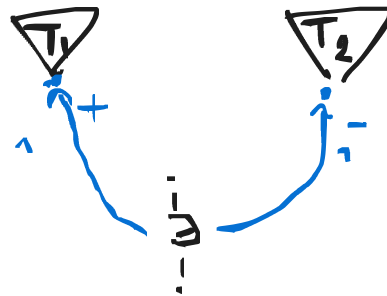
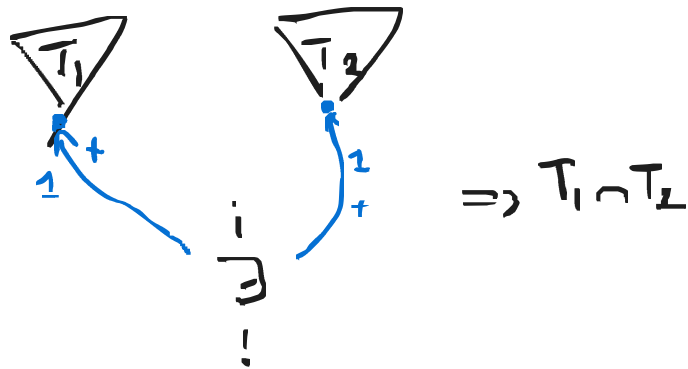
$T_1 \cup T_2$ iff $\exists S$: T_1 typ. A and T_2 typ. $A^\perp$ and



Coherence of Witnesses

$T_1 \cap T_2$ iff $\exists S$: T_1 and T_2 typeable by A and

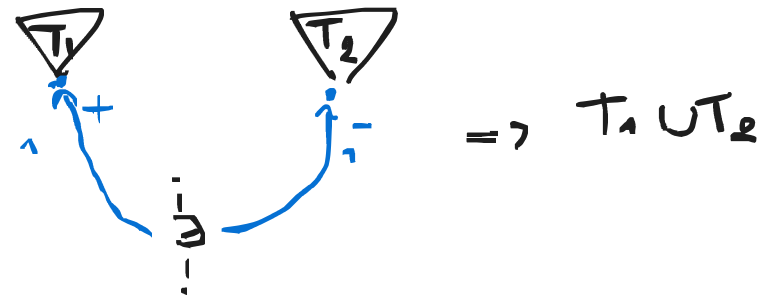
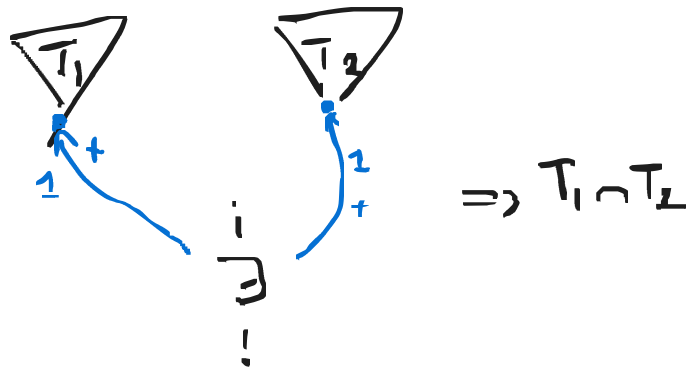
$T_1 \cup T_2$ iff $\exists S$: T_1 typ. A and T_2 typ. $A^\perp$ and



Coherence of Witnesses

$T_1 \cap T_2$ iff $\exists S : T_1$ and T_2 typeable by A and

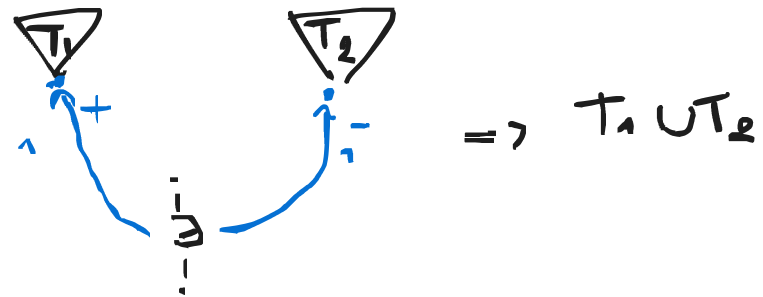
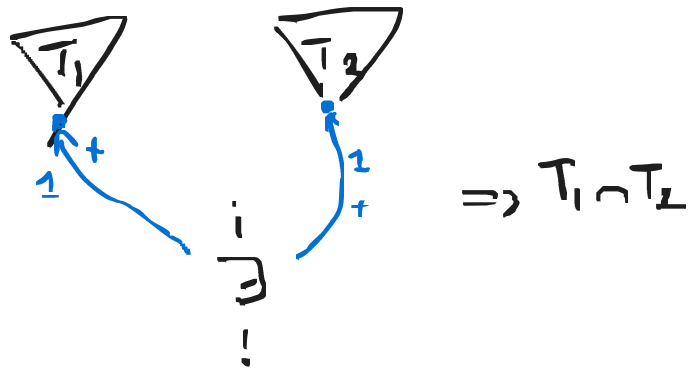
$T_1 \cup T_2$ iff $\exists S : T_1$ typ. A and T_2 typ. $A^\perp$ and



Coherence of Witnesses

$T_1 \cap T_2$ iff $\exists S$: T_1 and T_2 typeable by A and

$T_1 \cup T_2$ iff $\exists S$: T_1 typ. A and T_2 typ. $A^\perp$ and

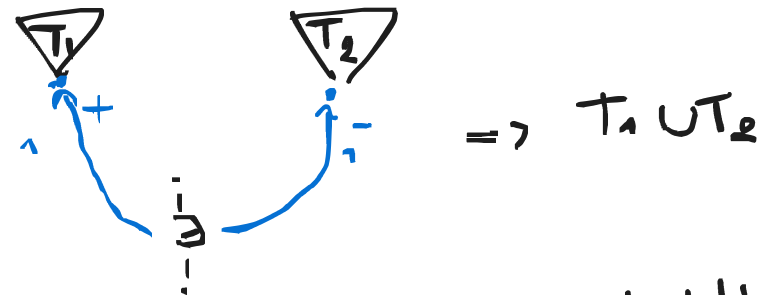
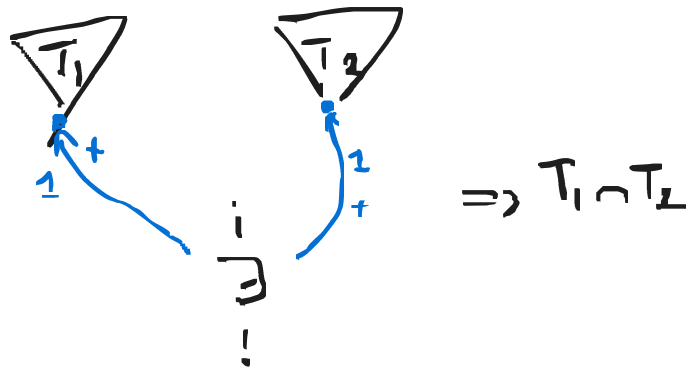


WE ASSUME the coherence of witnesses

Coherence of Witnesses

$T_1 \cap T_2$ iff $\exists S$: T_1 and T_2 typeable by A and

$T_1 \cup T_2$ iff $\exists S$: T_1 typ. A and T_2 typ. $A^\perp$ and



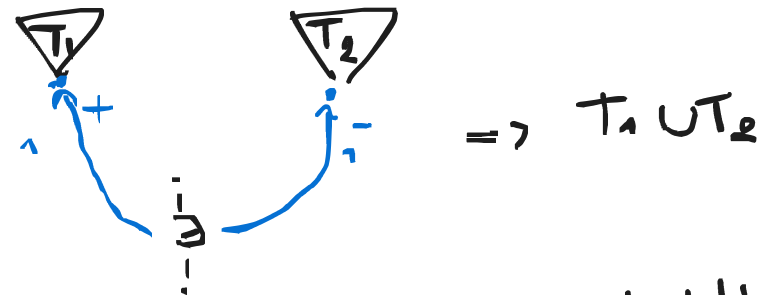
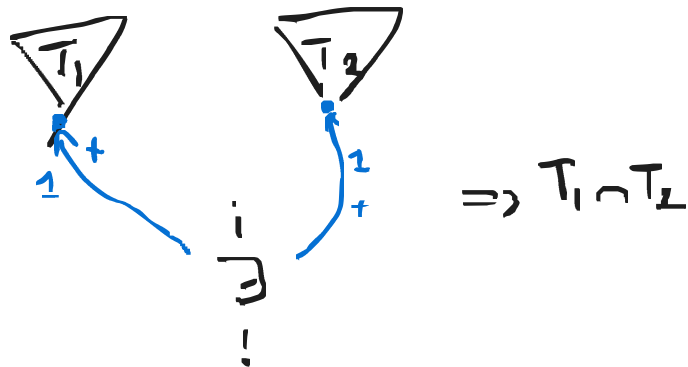
WE ASSUME the coherence of witnesses

testable in quadratic time.

Coherence of Witnesses

$T_1 \cap T_2$ iff $\exists S$: T_1 and T_2 typeable by A and

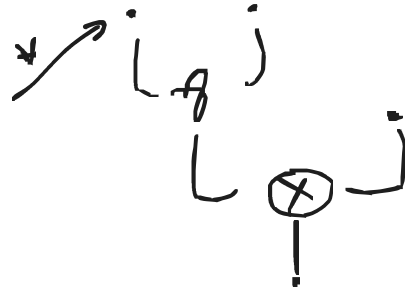
$T_1 \cup T_2$ iff $\exists S$: T_1 typ. A and T_2 typ. $A^\perp$ and



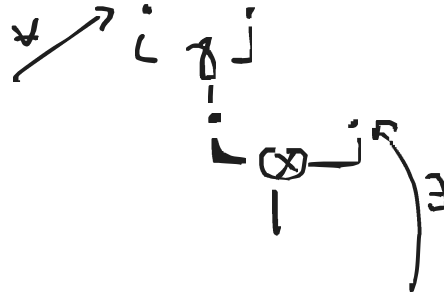
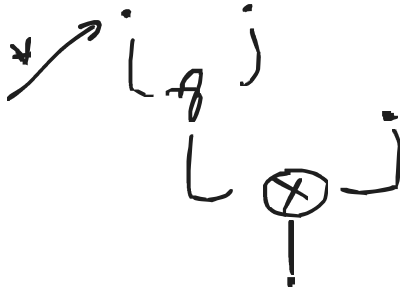
WE ASSUME the coherence of witnesses

testable in
quadratic time.
(read n-trees simultaneously)

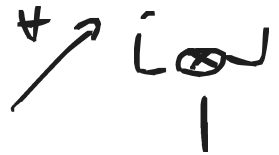
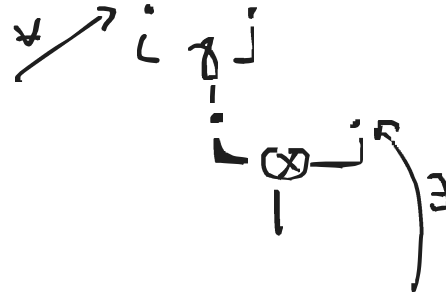
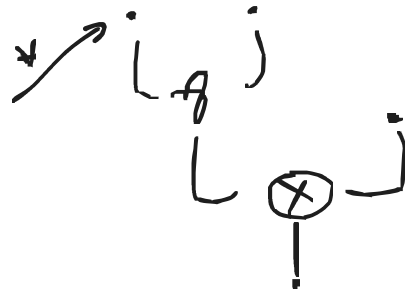
Coherence of Witnesses



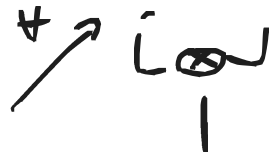
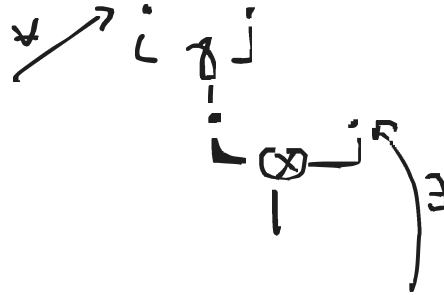
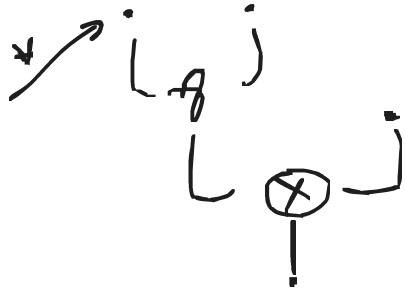
Coherence of Witnesses



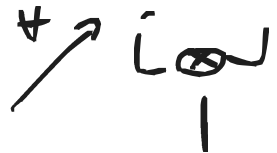
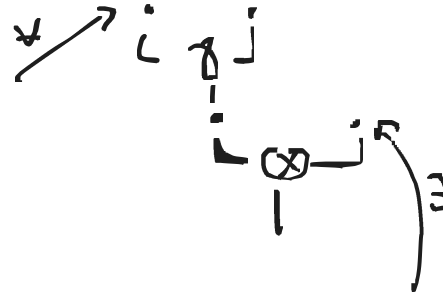
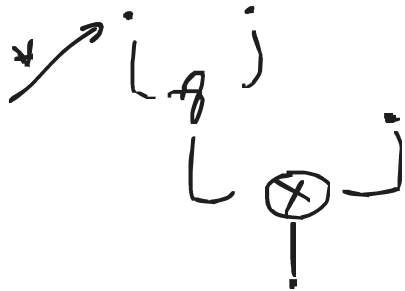
Coherence of Witnesses



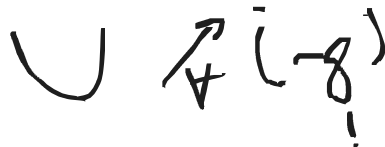
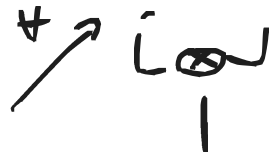
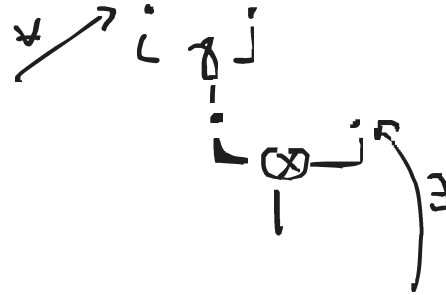
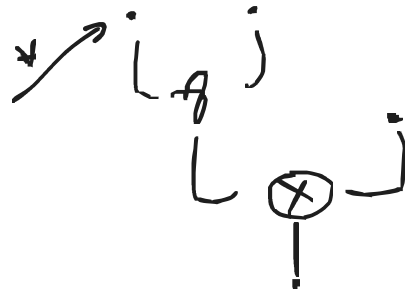
Coherence of Witnesses



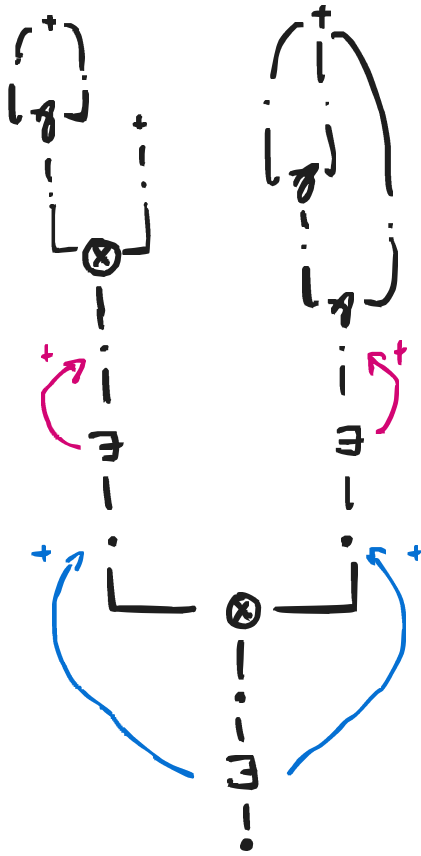
Coherence of Witnesses



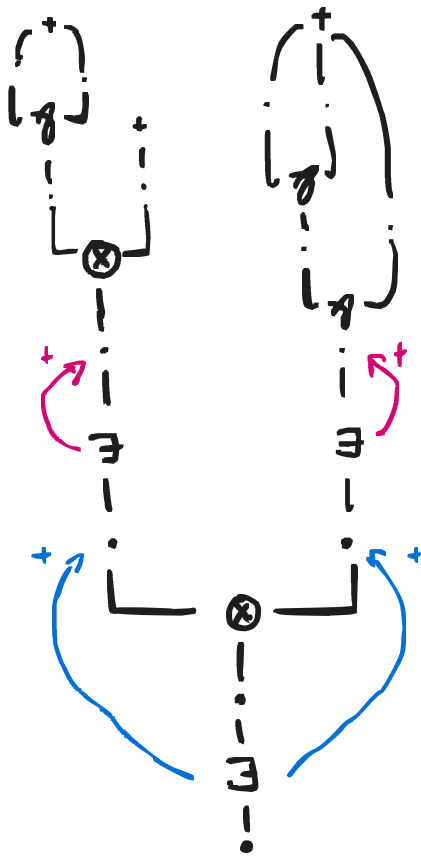
Coherence of Witnesses



Hidden Trees

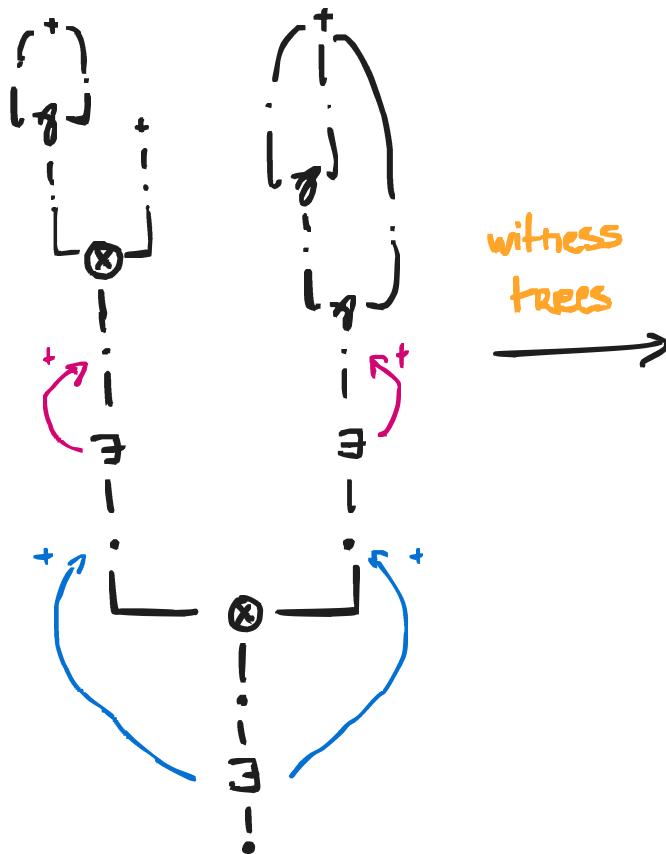


Hidden Trees



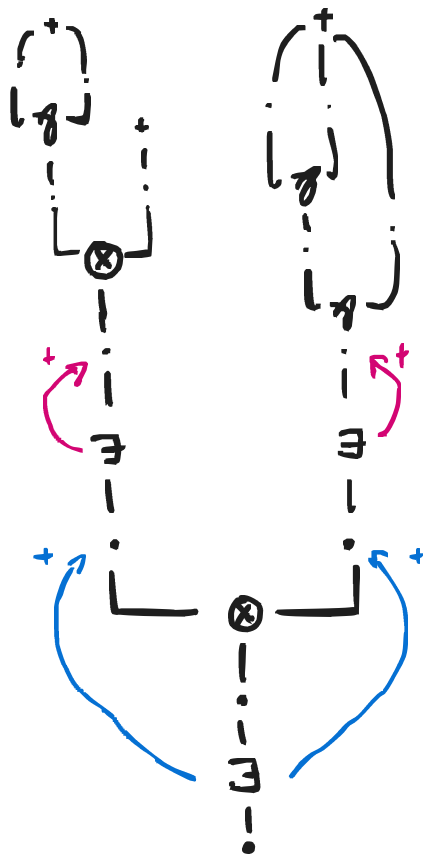
are the blue witnesses coherent?

Hidden Trees



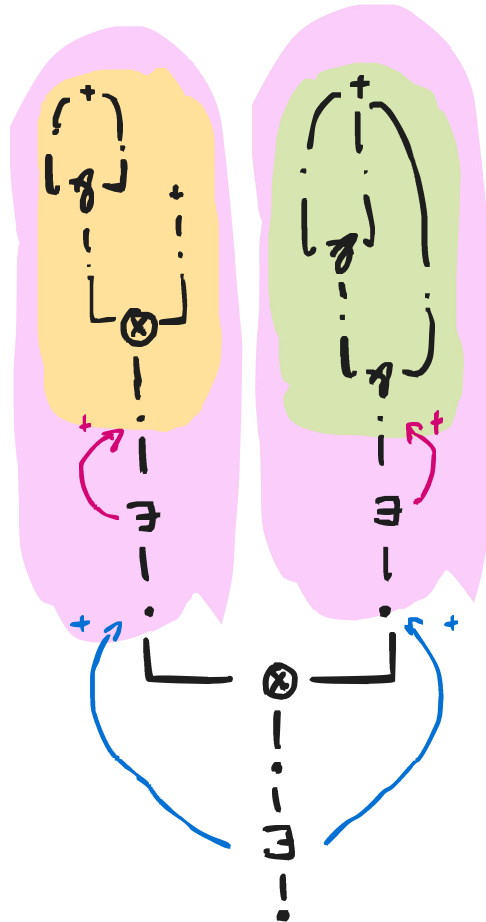
are the blue witnesses
coherent?

Hidden Trees

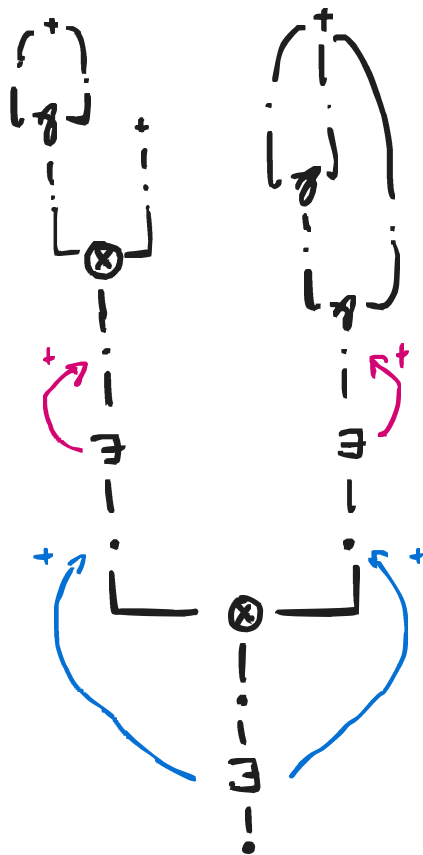


are the blue witnesses
coherent?

witness
trees

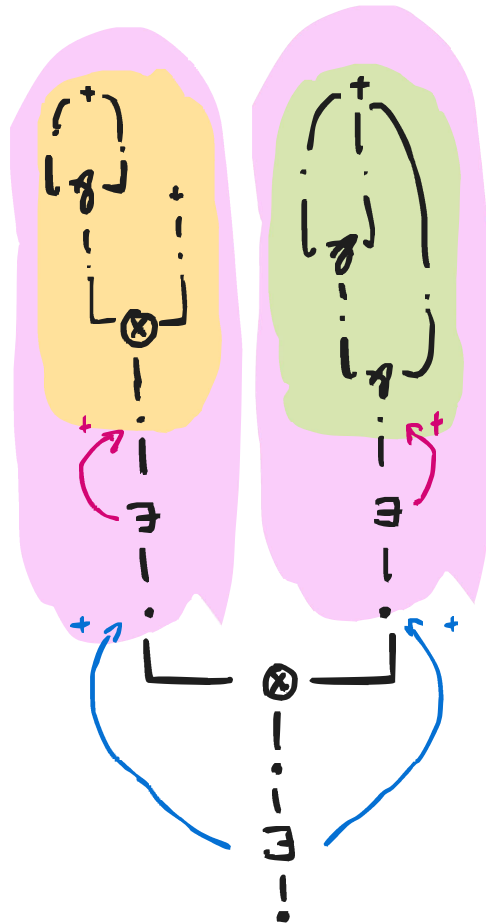


Hidden Trees



are the blue witnesses coherent?

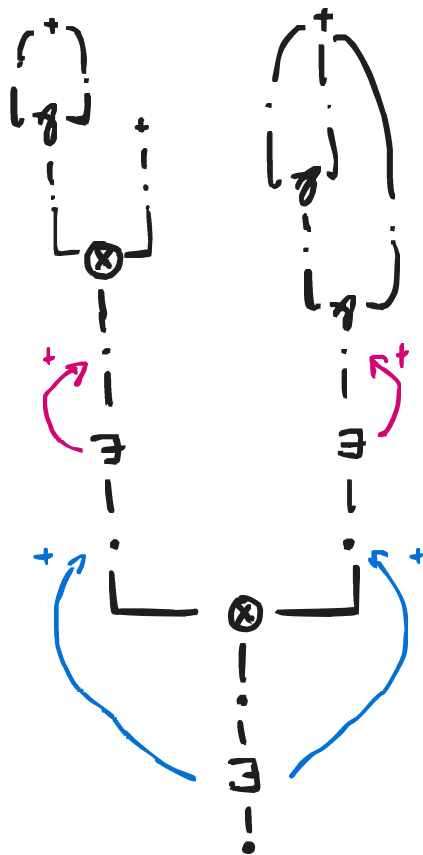
witness trees



erase hidden trees

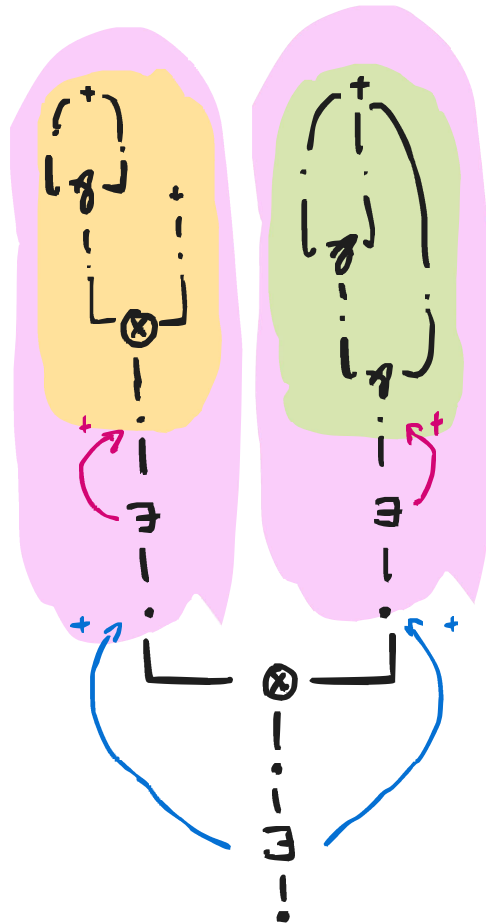


Hidden Trees

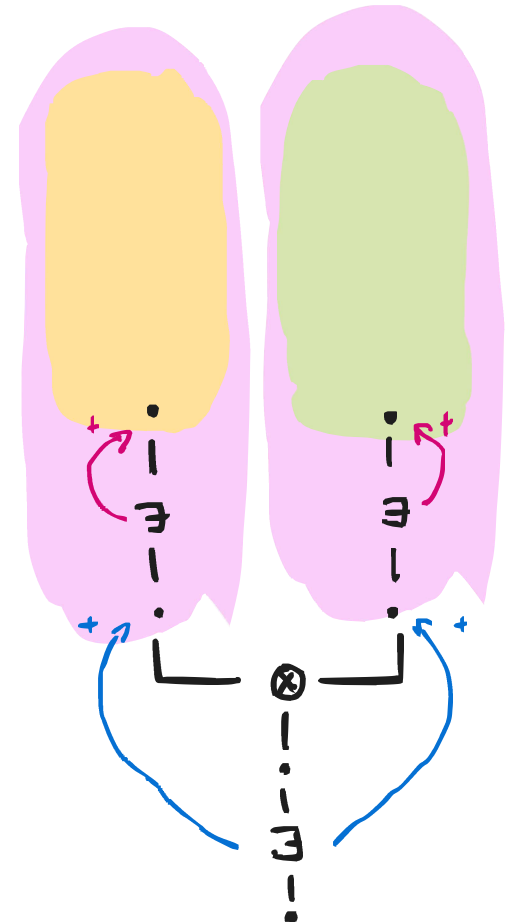


are the blue witnesses coherent?

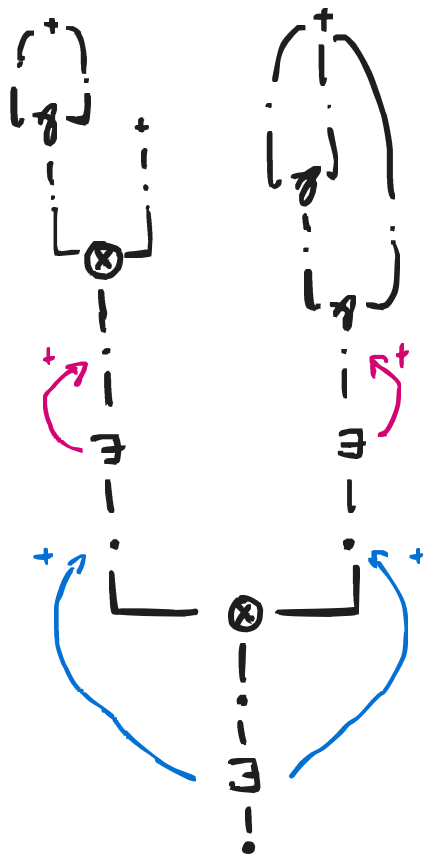
witness trees



erase
hidden trees

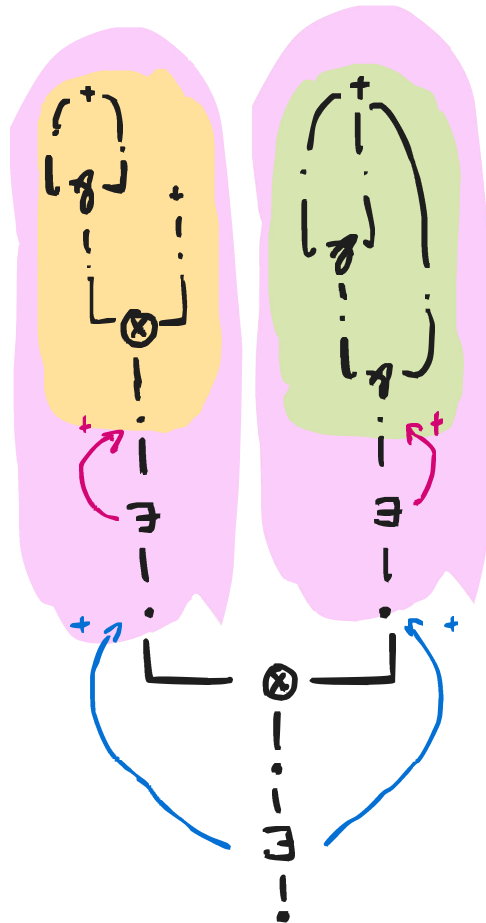


Hidden Trees

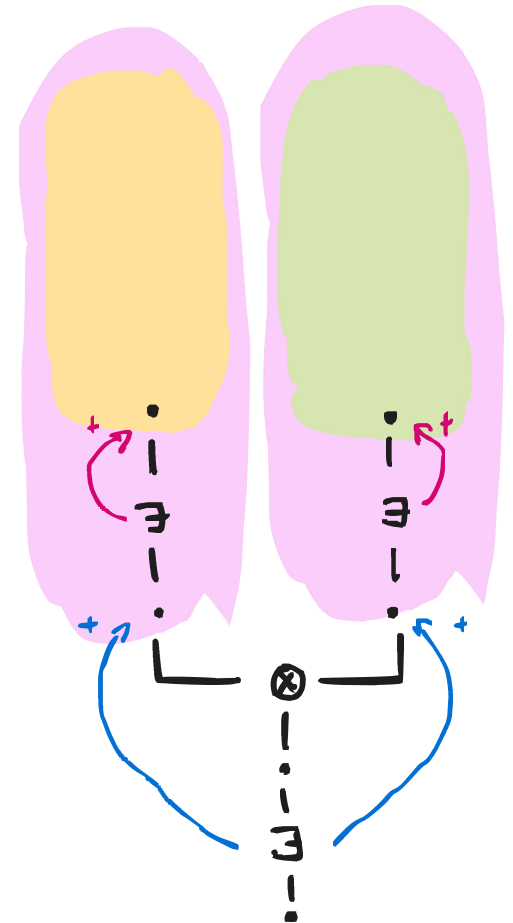


are the blue witnesses coherent?

witness trees



erase hidden trees



Yes! they are coherent!

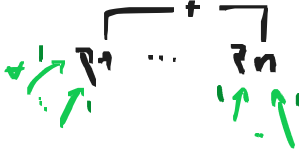
Desequentialisation

Desequentialisation (1/2)

Desequentialisation (1/2)

$$\langle \overline{A_1, \dots, A_n}^+, \text{VAR} \rangle$$

Desequentialisation (1/2)

$$\left\langle \overline{A_1, \dots, A_n}^+, \text{VAR} \right\rangle \longrightarrow \overline{P_1^+ \dots P_n^+}$$


Desequentialisation (1/2)

$$\left\langle \frac{}{A_1, \dots, A_n}^+, \text{VAR} \right\rangle$$

$\longrightarrow$

$$\frac{}{p_1 \dots p_n}^+$$

(Note: Green arrows in the original image point from the empty numerator to each p_i)

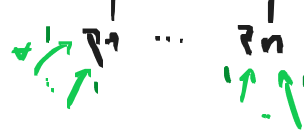
$$x \xrightarrow{\text{green}} p_i$$

iff
 $x \in \text{VAR}$
 $x \in \text{FV}(A_i)$

Desequentialisation (1/2)

$\langle \overline{A_1, \dots, A_n}^+, \text{VAR} \rangle$

$\longrightarrow$

$\overline{p_1 \dots p_n}^+$


$x \rightarrow p_i$
 iff
 $x \in \text{VAR}$
 $x \in \text{FV}(A_i)$

creates
 floating pointers

Desequentialisation (1/2)

$$\left\langle \overline{A_1, \dots, A_n}^+, \text{VAR} \right\rangle$$

$\longrightarrow$

$$\overline{p_1 \dots p_n}^+$$

$$\begin{aligned} x_i^+ &\rightarrow p_i \\ \text{iff} \\ x \in \text{VAR} \\ x \in \text{FV}(A_i) \end{aligned}$$

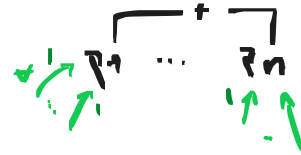
create
floating pointers

$$\left\langle \frac{\begin{matrix} \vdots \pi \\ T_1 A_1 A_2 \end{matrix}}{T_1 A_1 A_2} \gamma, \text{VAR} \right\rangle$$

Desequentialisation (1/2)

$$\left\langle \frac{\quad}{A_1, \dots, A_n} +, \text{VAR} \right\rangle$$

$\longrightarrow$



$x_i \rightarrow p_i$
 iff
 $x \in \text{VAR}$
 $x \in \text{FV}(A_i)$

create
 floating pointers

$$\left\langle \frac{\begin{matrix} \vdots \pi \\ \tau, A_1, A_2 \end{matrix}}{\tau, A_1, A_2} \gamma, \text{VAR} \right\rangle$$

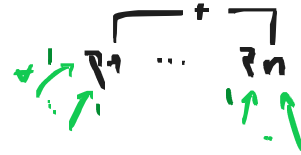
$\longrightarrow$



Desequentialisation (1/2)

$$\left\langle \frac{}{A_1, \dots, A_n} +, \text{VAR} \right\rangle$$

$\longrightarrow$



$x_i \rightarrow p_i$
 iff
 $x \in \text{VAR}$
 $x \in \text{FV}(A_i)$

create
 floating pointers

$$\left\langle \frac{\begin{matrix} \vdots \pi \\ \tau, A_1, A_2 \end{matrix}}{\tau, A_1, A_2} \gamma, \text{VAR} \right\rangle$$

$\longrightarrow$

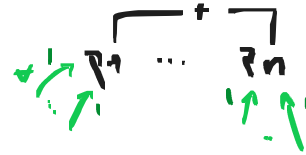


$$\left\langle \frac{\begin{matrix} \vdots \pi_1 & \vdots \pi_2 \\ \tau, A & D, B \end{matrix}}{\tau, A \otimes B, D} \otimes, \text{VAR} \right\rangle$$

Desequentialisation (1/2)

$$\left\langle \frac{}{A_1, \dots, A_n} +, \text{VAR} \right\rangle$$

→



$x_i \rightarrow p_i$
 iff
 $x \in \text{VAR}$
 $x \in \text{FV}(A_i)$

create
floating pointers

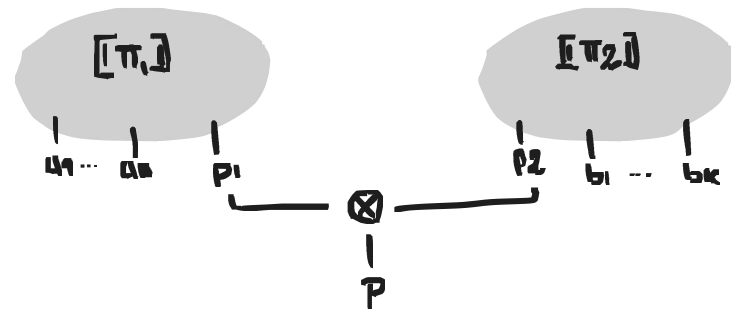
$$\left\langle \frac{\begin{matrix} \vdots \pi \\ \tau, A_1, A_2 \end{matrix}}{\tau, A_1, A_2} \gamma, \text{VAR} \right\rangle$$

→



$$\left\langle \frac{\begin{matrix} \vdots \pi_1 & \vdots \pi_2 \\ \tau, A & D, B \end{matrix}}{\tau, A \otimes B, D} \otimes, \text{VAR} \right\rangle$$

→

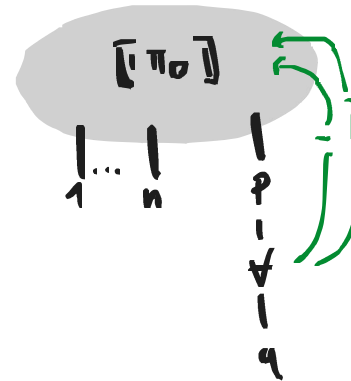


Desquentialisation (2/2)

$$\left\langle \frac{\overset{\vdots \tau_0}{T, A}}{T, V \times A} V, \text{VAR} \right\rangle$$

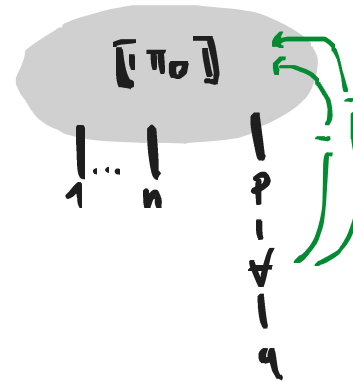
Desquentialisation (2/2)

$$\left\langle \frac{\overset{\vdots \pi_0}{T, A}}{T, V \times A} V, \text{VAR} \right\rangle \mapsto$$



Desquentialisation (2/2)

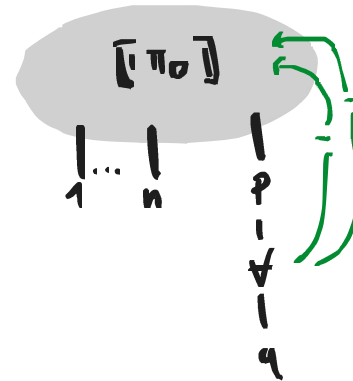
$$\left\langle \frac{\vdots \pi_0}{\Gamma, A} \forall, \text{VAR} \right\rangle \longrightarrow$$



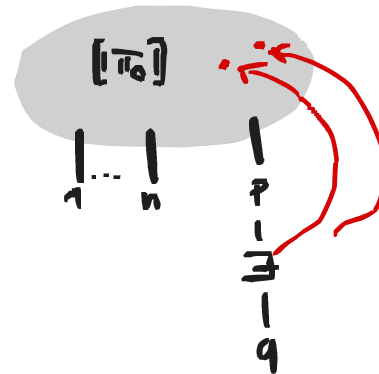
$$\left\langle \frac{\vdots \pi_0}{\Gamma, A [B/y]} \exists, \text{VAR} \right\rangle$$

Desquentialisation (2/2)

$$\left\langle \frac{\vdots \pi_0}{\Gamma, A} \forall, \text{VAR} \right\rangle \mapsto$$

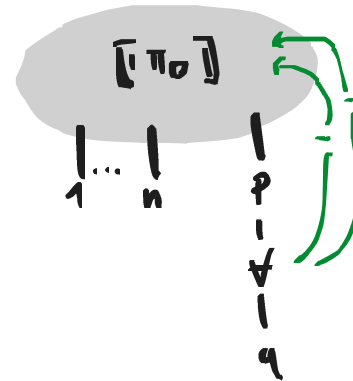


$$\left\langle \frac{\vdots \pi_0}{\Gamma, A[B/y]} \exists, \text{VAR} \right\rangle \mapsto$$

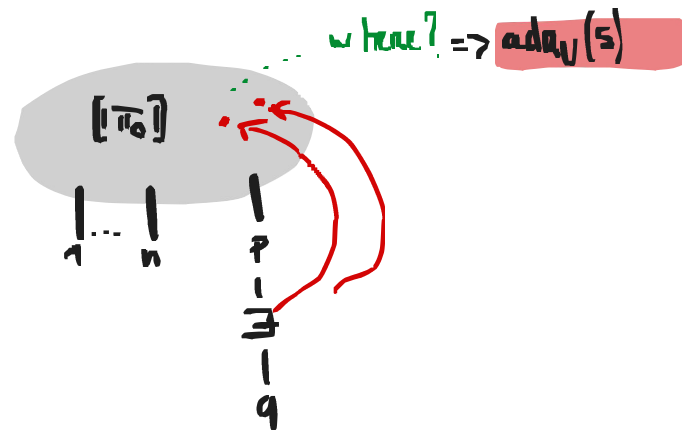


Desquentialisation (2/2)

$$\left\langle \frac{\vdots \pi_0}{\Gamma, A} \forall, \text{VAR} \right\rangle \mapsto$$

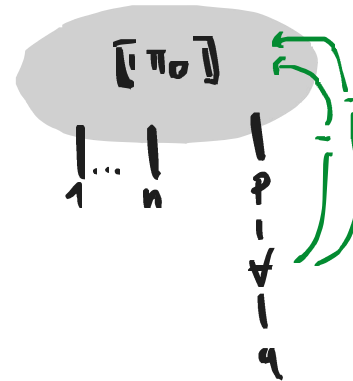


$$\left\langle \frac{\vdots \pi_0}{\Gamma, A [B/y]} \exists, \text{VAR} \right\rangle \mapsto$$

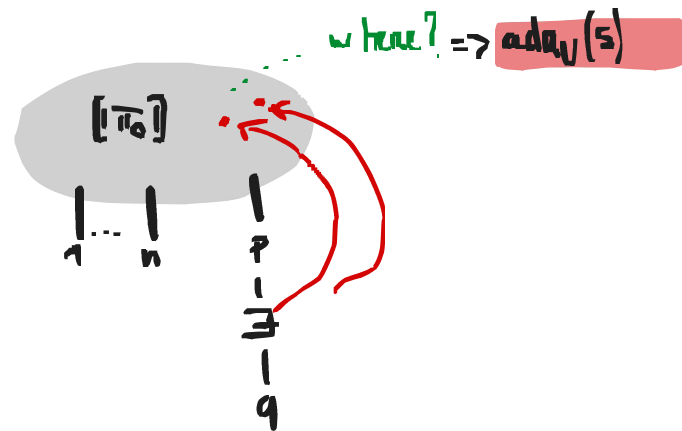


Desquentialisation (2/2)

$$\left\langle \frac{\vdots \pi_0}{\Gamma, A} \forall, \text{VAR} \right\rangle \mapsto$$



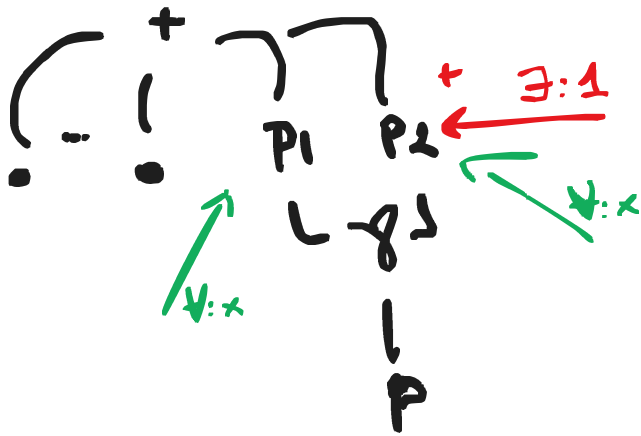
$$\left\langle \frac{\vdots \pi_0}{\Gamma, A [B/y]} \exists, \text{VAR} \right\rangle \mapsto$$



$$[\pi] \stackrel{\text{DEF}}{=} \langle \pi, \text{EigenVar}(\pi) \rangle$$

PARSING

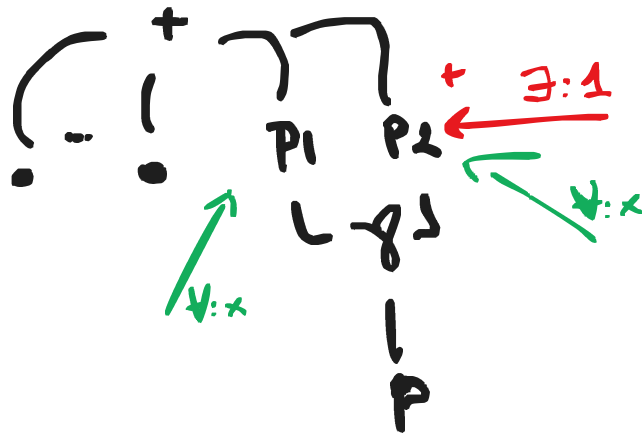
PARSING



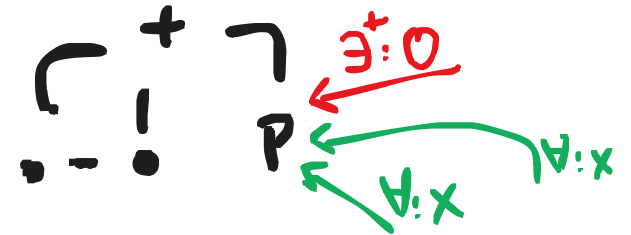
→ P_0 KILLS J -pointers

→ P_0 BLOCKED if alive J -pointer is not alone

PARSING



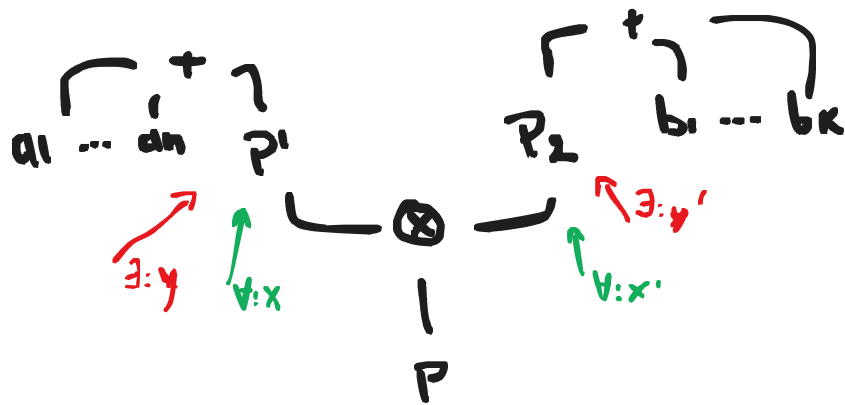
$\rightarrow \bar{P}$



$\rightarrow P_0$ KILLS E -pointers

$\rightarrow P_0$ BLOCKED if alive E -pointer is not alone

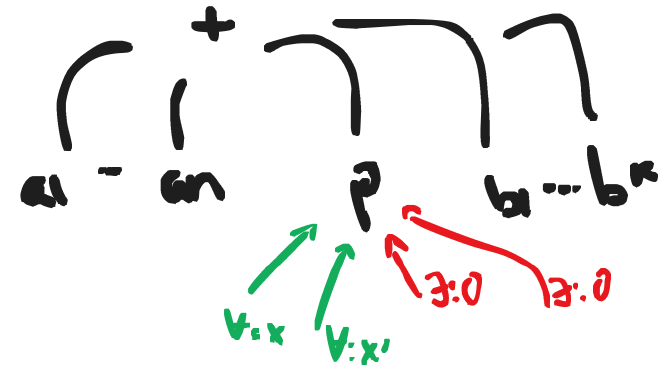
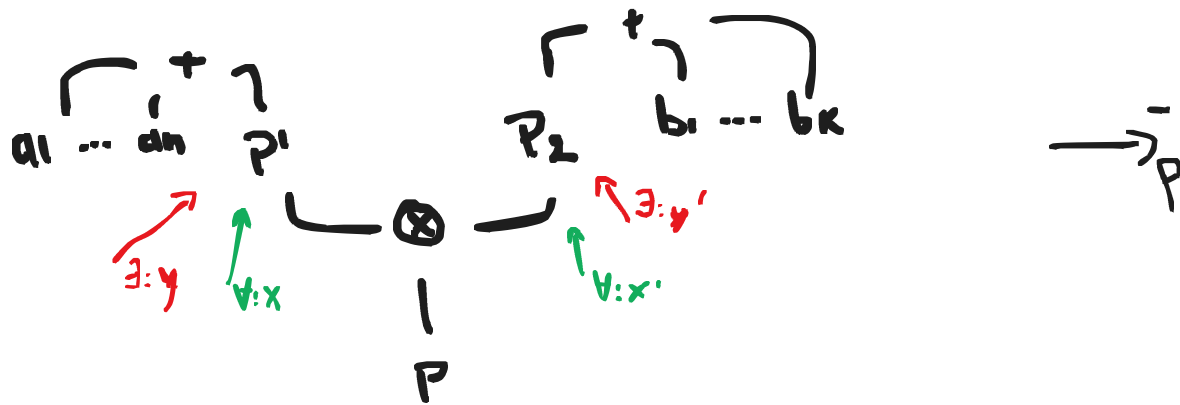
PARSING



$\rightarrow_{p_0}$ KILLS $\exists$ -pointers

$\rightarrow_{p_0}$ BLOCKED if alive $\exists$ -pointer is not alone

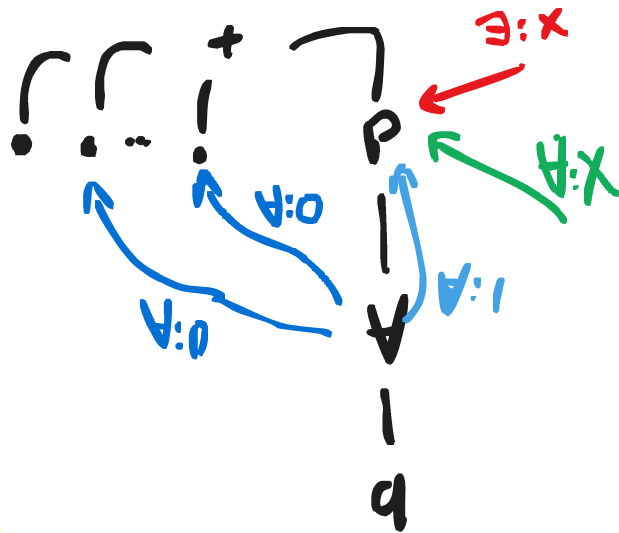
PARSING



$\rightarrow P_0$ KILLS $\exists$ -pointers

$\rightarrow P_0$ BLOCKED if alive $\exists$ -pointer is not alone

PARSING

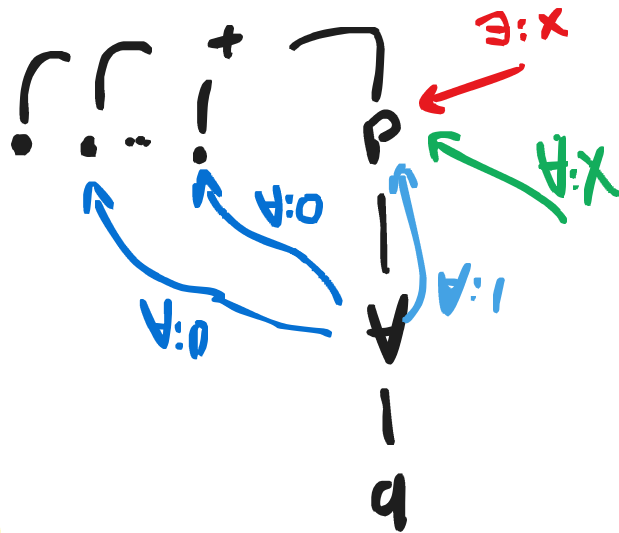


all pointers
are **dead**
and
in the
daimon above!

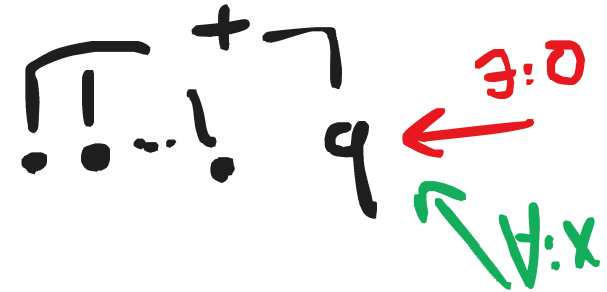
→ p_0 KILLS $\exists$. pointers

→ p_0 BLOCKED if alive $\exists$ -pointer is not alone

PARSING



$\rightarrow \bar{p}$

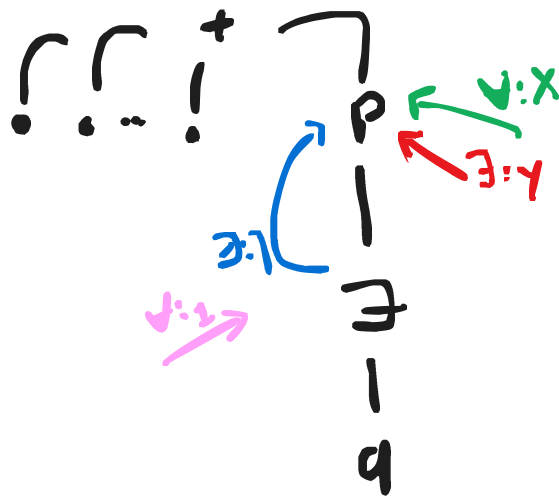


all pointers
are **dead**
and
in the
daimon above!

$\rightarrow p_0$ KILLS $\exists$. pointers

$\rightarrow p_0$ BLOCKED if alive $\exists$ -pointer is not alone

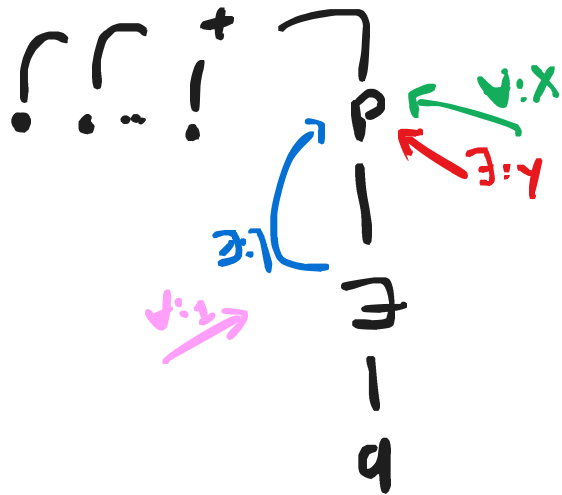
PARSING



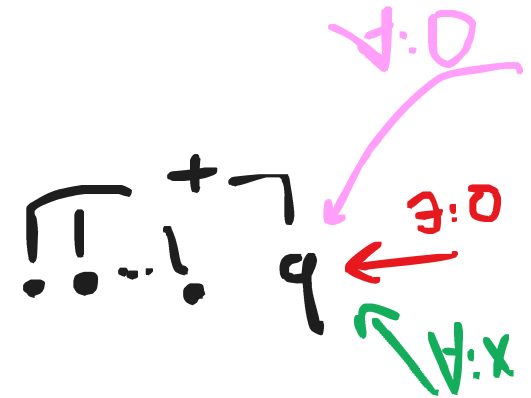
→ p_0 KILLS λ -pointers

→ p_0 BLOCKED if alive λ -pointer is not alone

PARSING



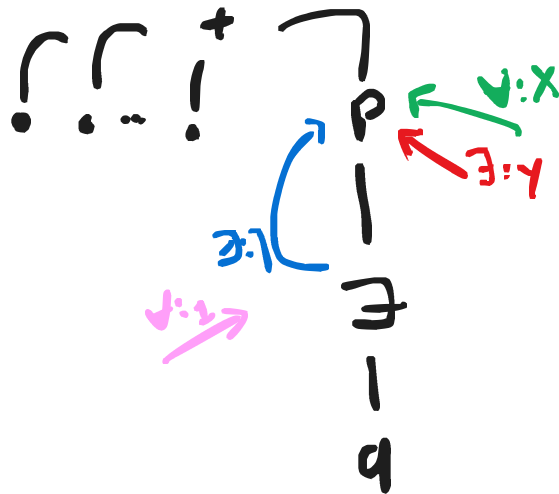
$\rightarrow \bar{p}$



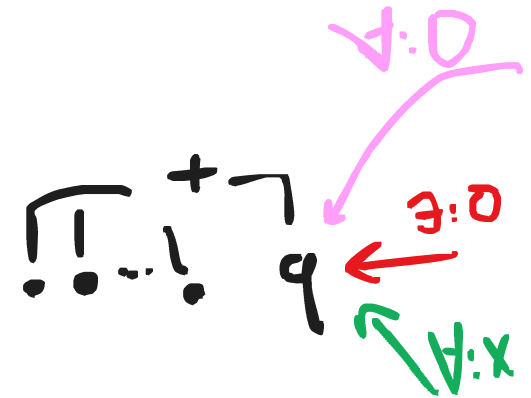
$\rightarrow p_0$ KILLS f -pointers

$\rightarrow p_0$ BLOCKED if alive f -pointer is not alone

PARSING



$\rightarrow_p$

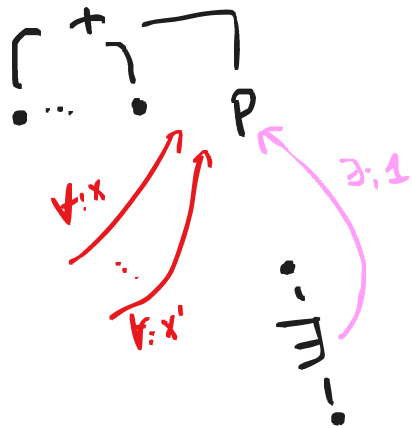


$\forall x$ pointers of the x -link
deactivates

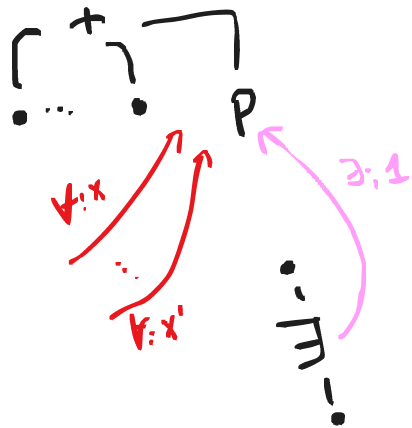
$\rightarrow_p$ KILLS x -pointers

$\rightarrow_p$ BLOCKED if alive x -pointer is not alone

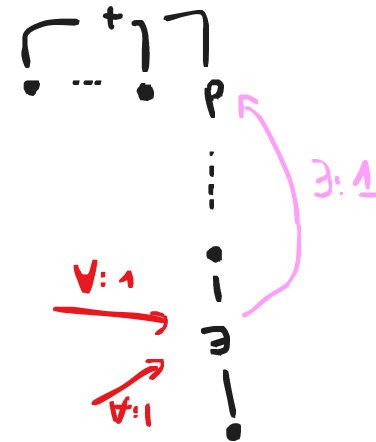
PARSING: Pointer Rule



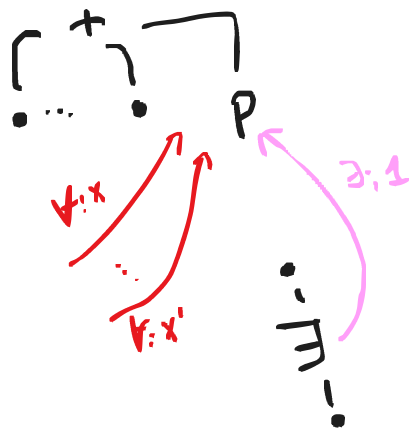
PARSING: Pointer Rule



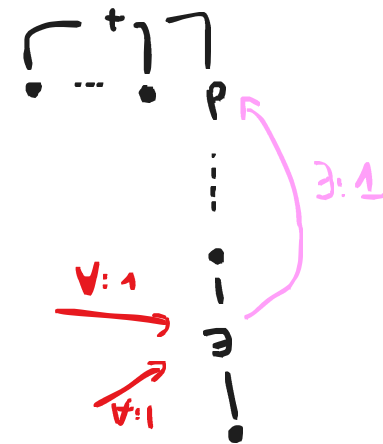
$\rightarrow P_{tr}$



PARSING: Pointer Rule



→ Ptr



alive V-pointers
are **rerouted**
to the 3-link.

PARSING

$$\rightarrow_P = \rightarrow_{\bar{P}} \cup \rightarrow_{PTR}$$

(but always perform $\rightarrow_{PTR}$ first)

so $\rightarrow_P = \rightarrow_{PTR} \cdot \rightarrow_{\bar{P}}$

PLAN

I MLL2 Nets : challenges

II Typed vs untyped nets

- OUR CRITERION

III Nets : desequentializing proofs

- OUR CRITERION

IV PARSING strategy

V PARSING (naive)

VI PARSING with schemes

PLAN

I MLL2 Nets : challenges

II Typed vs untyped nets

- OUR CRITERION

III Nets : desequentializing proofs

- OUR CRITERION 

IV PARSING strategy

V PARSING (naive)

VI PARSING with schemes

MAIN RESULT

$$S = [\pi]$$

↑ proof of ML2

$\Leftrightarrow$

$$S \xrightarrow[p]{*} \begin{bmatrix} * \\ 1 \dots n \end{bmatrix}$$

MAIN RESULT

$$S = [\pi]$$

↑ proof of $ML2^\pm$

$\Leftrightarrow$

$$S \xrightarrow[p]{*} \begin{bmatrix} * \\ 1 \dots n \end{bmatrix}$$

MAIN RESULT

SOUNDNESS

$$S = [\pi]$$

↑ proof of ML2

$\Rightarrow$

$$S \xrightarrow[p]{*} \begin{bmatrix} \vdots \\ 1 \dots n \end{bmatrix}$$

MAIN RESULT

SOUNDNESS

$$S = [\pi] \quad \Rightarrow \quad S \xrightarrow[p]{*} \begin{bmatrix} * \\ 1 \dots n \end{bmatrix}$$

$\uparrow$ proof of ML2

COMPLETENESS

$$S \xrightarrow[p]{*} \begin{bmatrix} * \\ 1 \dots n \end{bmatrix} \quad \Rightarrow \quad S = [\pi]$$

$\uparrow$ proof of ML2

MAIN RESULT

SOUNDNESS


$$S = [\pi] \quad \Rightarrow \quad S \xrightarrow[p]{*} \begin{bmatrix} \vdots \\ 1 \dots n \end{bmatrix}$$

$\uparrow$ proof of ML2

COMPLETENESS

$$S \xrightarrow[p]{*} \begin{bmatrix} \vdots \\ 1 \dots n \end{bmatrix} \quad \Rightarrow \quad S = [\pi]$$

$\uparrow$ proof of ML2

MAIN RESULT

SOUNDNESS

$$S = [\pi]$$

↑ proof of ML2

$\Rightarrow$

$$S \rightarrow_p^* \begin{bmatrix} \vdots \\ 1 \dots n \end{bmatrix}$$

COMPLETENESS

$$S \rightarrow_p^* \begin{bmatrix} \vdots \\ 1 \dots n \end{bmatrix}$$

$\Rightarrow$

$$S = [\pi]$$

↑ proof of ML2

MAIN RESULT

SOUNDNESS

$$S = [\pi]$$

↑ proof of ML2

$\Rightarrow$

$$S \rightarrow_p^* \begin{bmatrix} \pi \\ 1 \dots n \end{bmatrix}$$

COMPLETENESS

$$S \rightarrow_p^* \begin{bmatrix} \pi \\ 1 \dots n \end{bmatrix}$$

$\Rightarrow$

$$S = [\pi]$$

↑ proof of ML2

PLAN

I MLL2 Nets : challenges

II Typed vs untyped nets

- OUR CRITERION

III Nets : desequentializing proofs

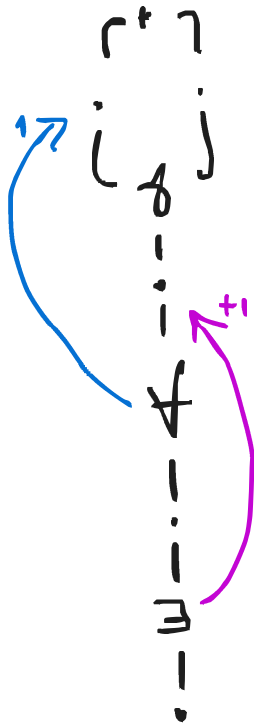
- OUR CRITERION

IV PARSING strategy ←

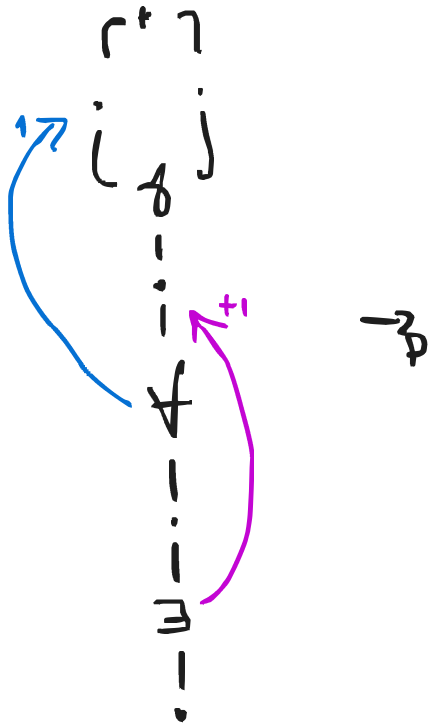
V PARSING (naive)

VI PARSING with schemes

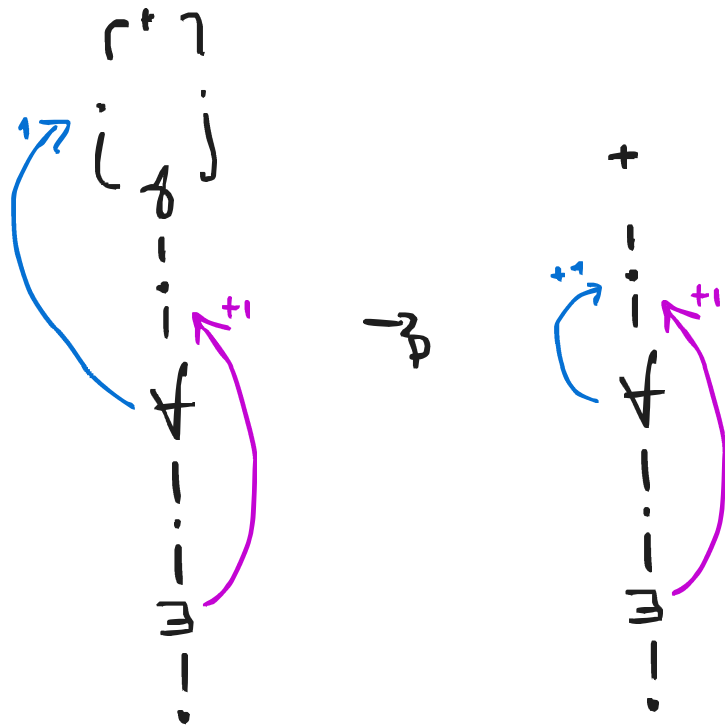
Reroute first



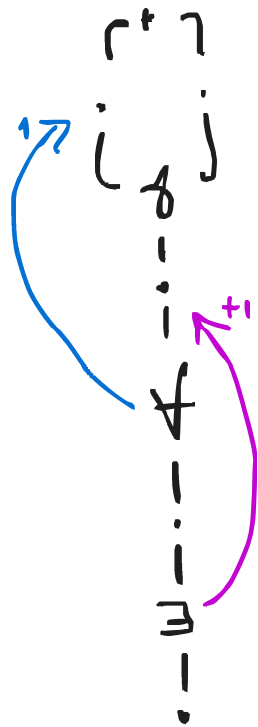
Reroute first



Reroute first



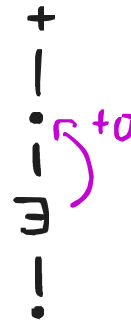
Reroute first



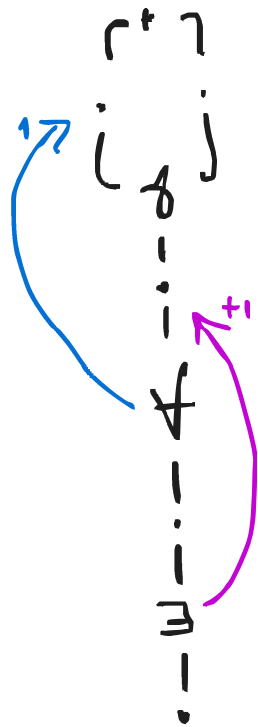
$\rightarrow$



$\rightarrow$



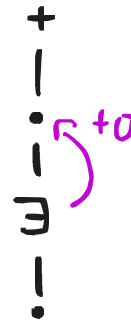
Reroute first



$\rightarrow$



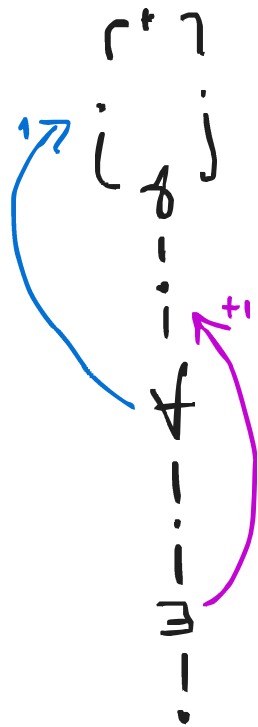
$\rightarrow$



$\rightarrow$



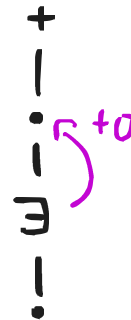
Reroute first



$\rightarrow$



$\rightarrow$

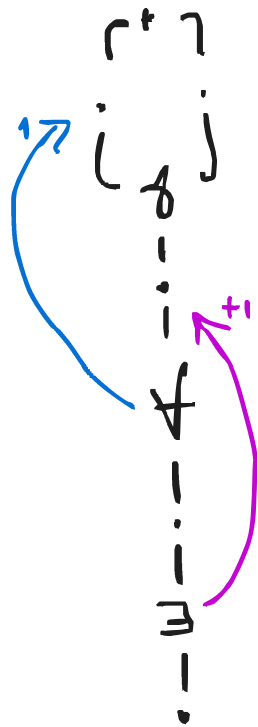


$\rightarrow$

$\vdots$

it is a proof net.

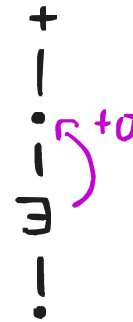
Reroute first



$\rightarrow$



$\rightarrow$



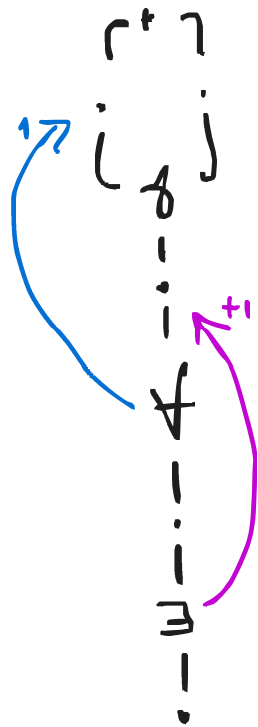
$\rightarrow$

$\vdots$

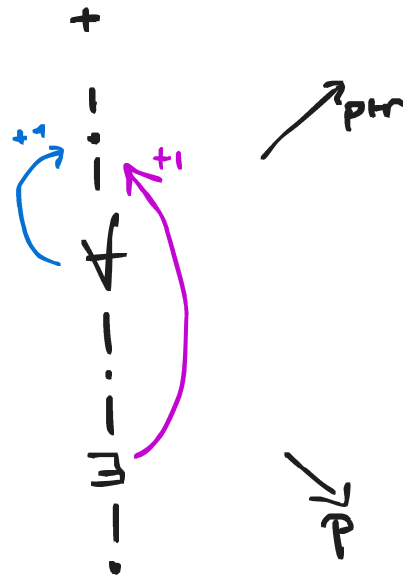


it is a proof net.

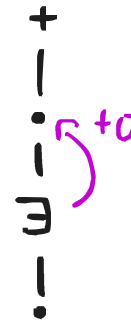
Reroute first



$\rightarrow$



$\rightarrow$

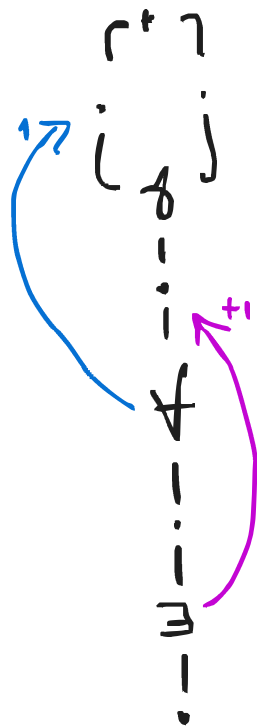


$\rightarrow$



 it is a proof net.

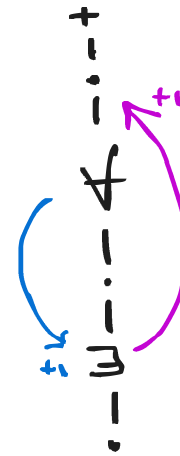
Reroute first



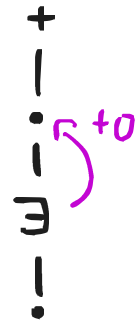
$\rightarrow$



$\nearrow_{ptr}$



$\searrow_p$



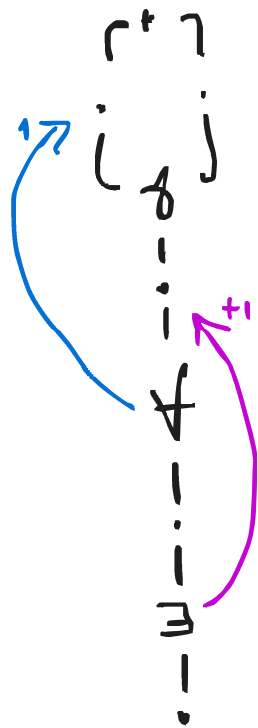
$\rightarrow_p$

$\vdots$



it is a proof net.

Reroute first



$\rightarrow$

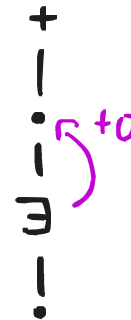


$\rightarrow$ ptr



$\rightarrow$?
P

$\rightarrow$ P



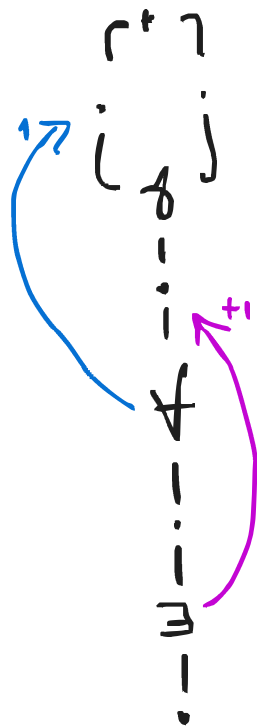
$\rightarrow$ P

$\rightarrow$



it is a proof net.

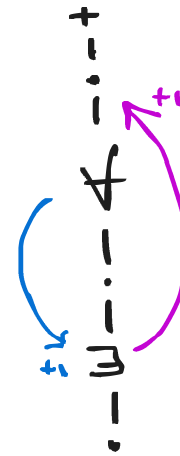
Reroute first



$\rightarrow$

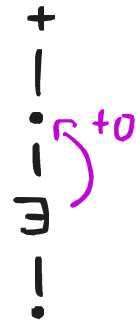


$\rightarrow$ p_{tr}



~~$\rightarrow$ p~~

$\rightarrow$ p

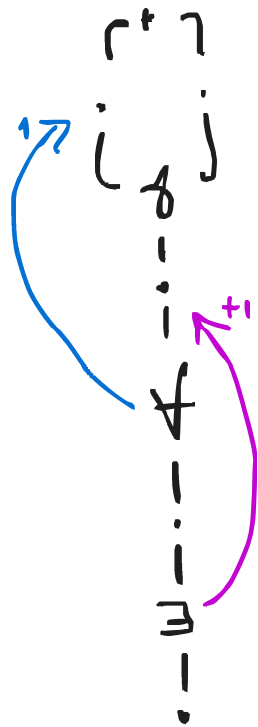


$\rightarrow$ p



it is a proof net.

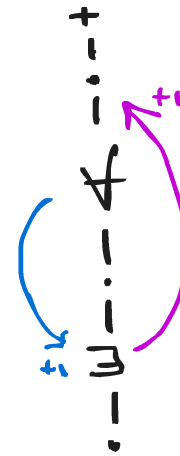
Reroute first



$\rightarrow$



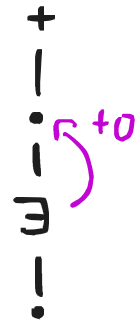
$\nearrow_{ptr}$



$\times \rightarrow_P$

not a proof net!

$\searrow_P$



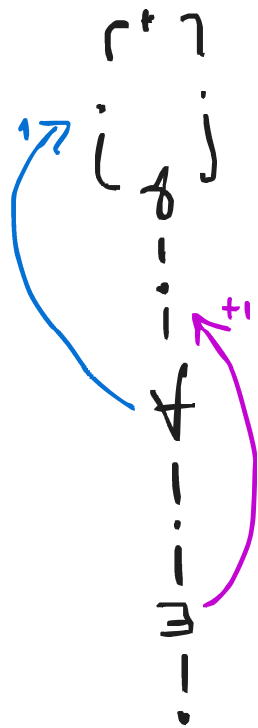
$\rightarrow_P$

$\vdots$



it is a proof net.

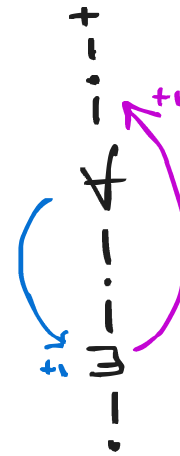
Reroute first



$\rightarrow$



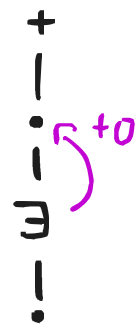
$\rightarrow$ p_{tr}



$\times$ $\rightarrow$ p

✓ not a proof net!

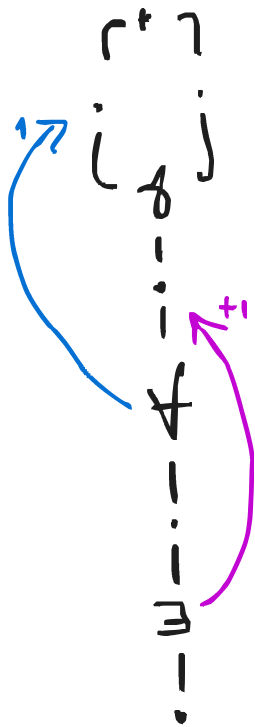
$\rightarrow$ p



$\rightarrow$ p

⚠ it is a proof net.

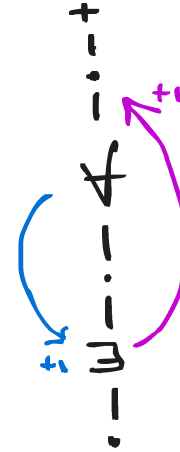
Reroute first



$\rightarrow$



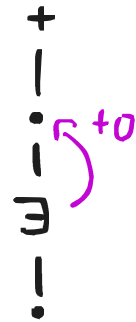
$\rightarrow$ p_{tr}



$\times$ $\rightarrow$ p

✓ not a proof net!

$\rightarrow$ p



$\rightarrow$ p

$\vdots$



it is a proof net.

CUTS AND SECOND
ORDER EXISTENTIAL
QUANTIFIERS -

ABOUT CUTS

Multiplicative
World -

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \longrightarrow \frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta, A \otimes A^\perp} \text{ (2)}$$

ABOUT CUTS

Multiplicative
World -

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \longrightarrow \frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta, A \otimes A^\perp} \otimes$$

$$\lfloor \text{cut} \rfloor \longrightarrow \lfloor \otimes \rfloor$$

ABOUT CUTS

Multiplicative
World -

+ 2nd Order
Quantifiers

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \longrightarrow$$

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta, A \otimes A^\perp} \otimes \quad ?$$

$\lfloor \text{cut} \rfloor$

$\longrightarrow$

$\lfloor \otimes \rfloor$
|

ABOUT CUTS

Multiplicative
World -

+ 2nd Order
Quantifiers

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \rightarrow$$

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta, A \otimes A^\perp} \otimes$$

X

$\lfloor \text{cut} \rfloor$

$\rightarrow$

$\lfloor \otimes \rfloor$
|

ABOUT CUTS

Multiplicative
World -

+ 2nd Order
Quantifiers

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \longrightarrow$$

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta, A \otimes A^\perp} \text{ (X)}$$

cuts and $\forall$ interact!

ABOUT CUTS

Multiplicative
World -

+ 2nd Order
Quantifiers

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \longrightarrow \frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta, A \otimes A^\perp} \otimes \quad \text{X}$$

cuts and $\forall$ interact!

$$\frac{\frac{\frac{\overline{x, x^\perp}^{\text{ax}} \quad \overline{x, x^\perp}^{\text{ax}}}{\text{cut}}}{x, x^\perp} \exists}{x, \exists w w} \forall}{w w \exists, \exists w w}$$

ABOUT CUTS

Multiplicative
World -

+ 2nd Order
Quantifiers

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \rightarrow$$

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta, A \otimes A^\perp} \otimes$$

X

cuts and $\forall$ interact!

$$\frac{\frac{\frac{\overline{x, x^\perp}^{\text{ax}} \quad \overline{x, x^\perp}^{\text{ax}}}{\text{cut}}}{x, x^\perp}^{\exists}}{x, \exists w w}^{\forall} \quad \frac{}{w w \exists z z}^{\forall}$$

ABOUT CUTS

Multiplicative
World -

+ 2nd Order
Quantifiers

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \longrightarrow$$

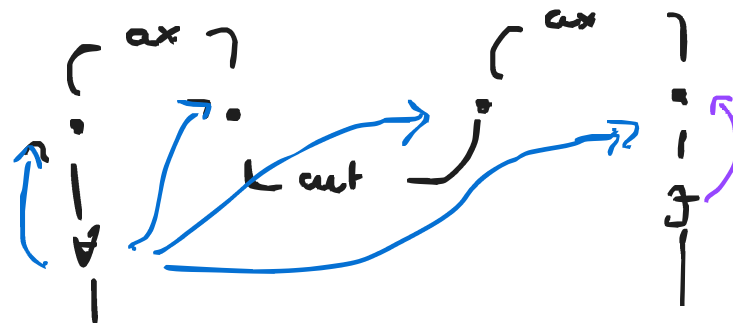
$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta, A \otimes A^\perp} \otimes$$

X

cuts and $\forall$ interact!

$$\frac{\frac{\frac{\frac{\overline{x, x^\perp}^{\text{ax}}}{x, x^\perp} \quad \frac{\overline{x, x^\perp}^{\text{ax}}}{x, x^\perp}}{x, x^\perp} \text{ cut}}{x, \exists w w} \exists}{\forall z z, \exists w w} \forall$$

$\mapsto$



ABOUT CUTS

cut $\sim \exists$?

ABOUT CUTS

cut $\sim \exists$ yes!

ABOUT CUTS

cut $\sim \exists$ **yes!**

$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut}$$

$\longrightarrow$

$$\frac{\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, A \otimes A^\perp, \Delta} \otimes}{\Gamma, \exists x x \otimes x^\perp, \Delta} \exists$$

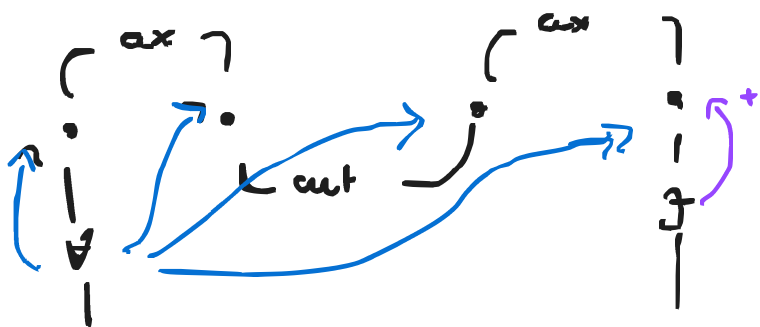
ABOUT CUTS

cut $\sim \exists$ yes!

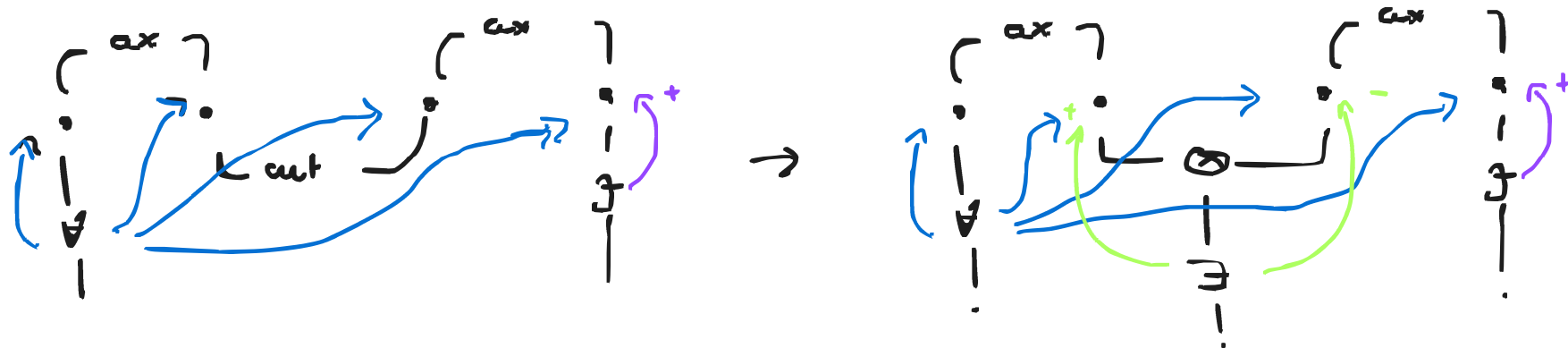
$$\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{ cut} \longrightarrow \frac{\frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, A \otimes A^\perp, \Delta} \otimes}{\Gamma, \exists x x \otimes t, \Delta} \exists$$

$\hookrightarrow$ cut

ABOUT CUTS



ABOUT CUTS

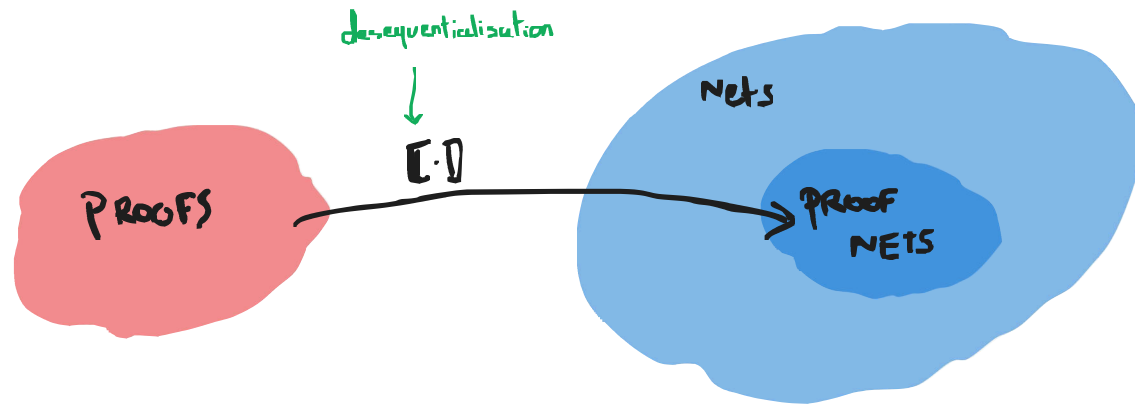


II - Parsing Strategy

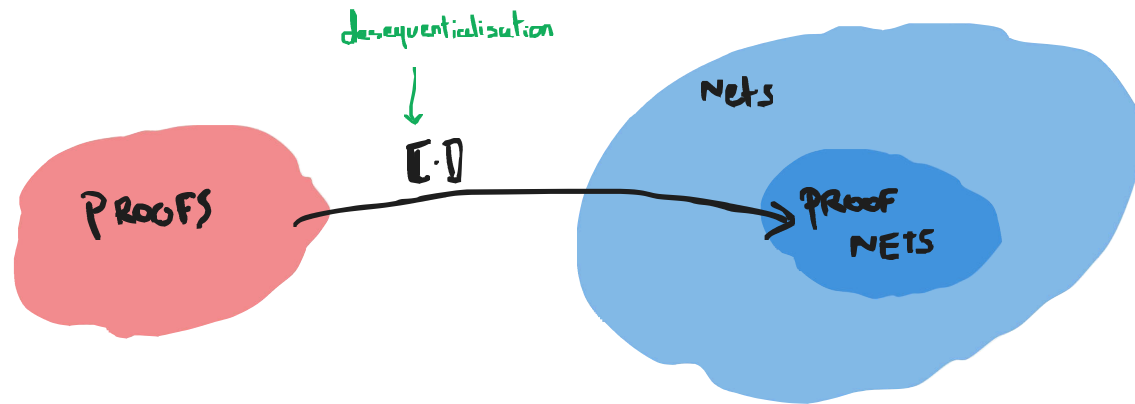
(for soundness
+ completeness)

PARSING
CRITERION using Diagrams
(PLAN)

PARSING situation

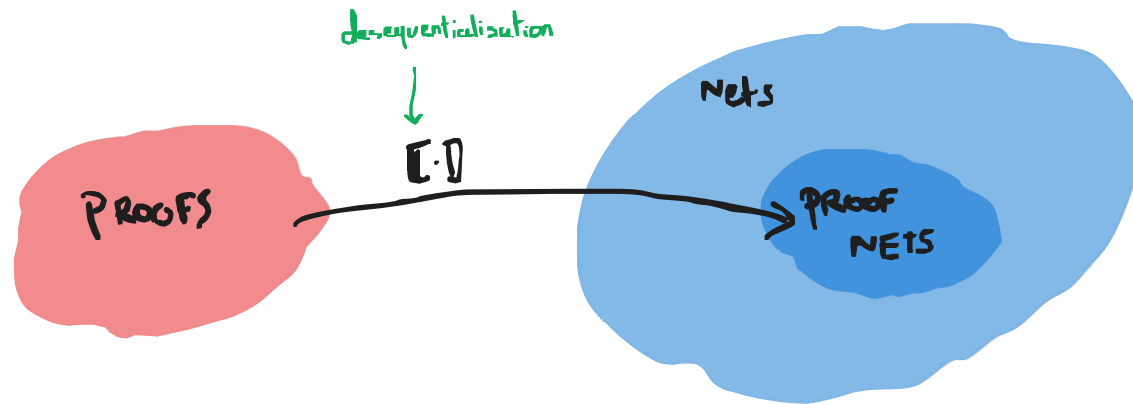


PARSING situation



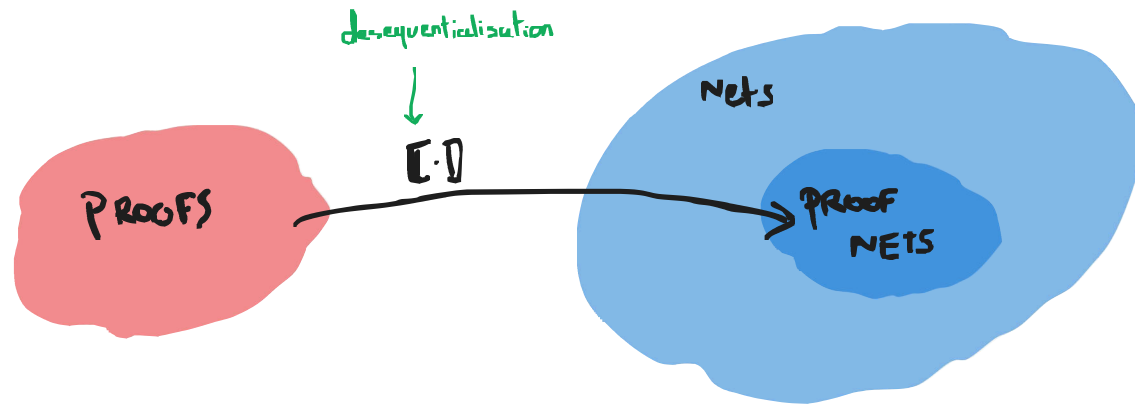
— α map $[.] : \text{proof} \rightarrow \text{nets}$

PARSING situation



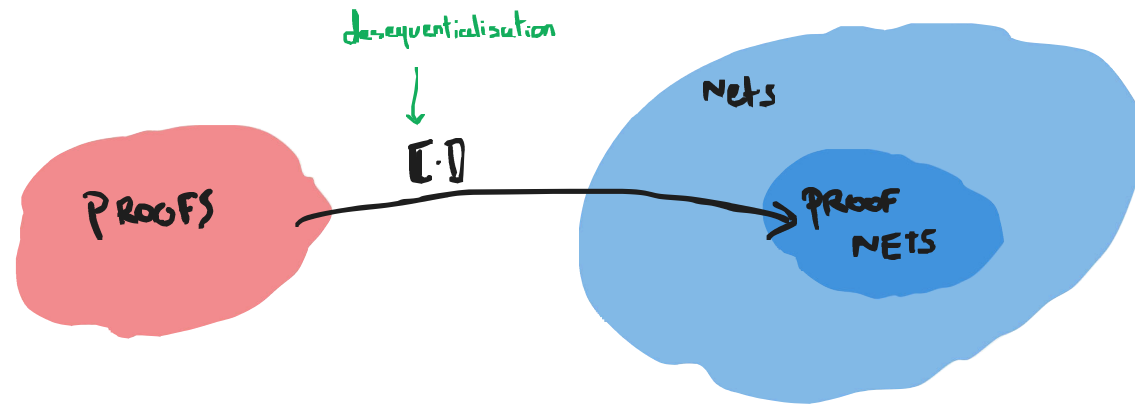
- a map $[.] : \text{proof} \rightarrow \text{nets}$
- a proof-search rewriting $\rightarrow_s$
- ..

PARSING situation



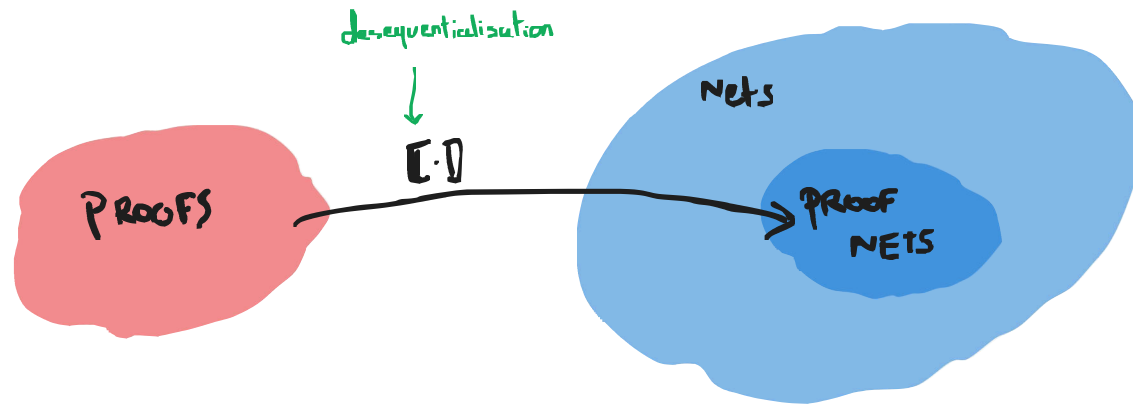
- a map $[.] : \text{proof} \rightarrow \text{nets}$
- a proof-search rewriting $\rightarrow_s$
- a rewriting on Nets $\rightarrow_p$

PARSING situation



- a map $[.] : \text{proof} \rightarrow \text{nets}$
- a proof-search rewriting $\rightarrow_s$
- a rewriting on Nets $\rightarrow_p$
 $\uparrow$
 PARSING

PARSING situation



- a map $[.] : \text{proof} \rightarrow \text{nets}$
- a proof-search rewriting $\rightarrow_s$
- a rewriting on Nets $\rightarrow_p$
 - $\uparrow$
 - PARSING
 - satisfying some diagrams