

Deep Inference for Graphical Theorem Proving

Pablo Donato

2025-06-10

Grothendieck Institute

D3S seminar

Prague

Introduction

$$\frac{x \text{ is } A \quad \text{All } A \text{ are } B}{x \text{ is } B} \rightsquigarrow \frac{A(x) \quad \forall y. A(y) \Rightarrow B(y)}{B(x)}$$

- *Generic patterns of deduction* as symbolic rules
-

$$\frac{x \text{ is } A \quad \text{All } A \text{ are } B}{x \text{ is } B} \rightsquigarrow \frac{A(x) \quad \forall y. A(y) \Rightarrow B(y)}{B(x)}$$

- *Generic patterns* of **deduction** as symbolic rules
- Formalist school (Hilbert):

Maths as a huge **game**

Goal: to prove theorems by following inference rules

$$\frac{x \text{ is } A \quad \text{All } A \text{ are } B}{x \text{ is } B} \rightsquigarrow \frac{A(x) \quad \forall y. A(y) \Rightarrow B(y)}{B(x)}$$

- *Generic patterns* of **deduction** as symbolic rules
- Formalist school (Hilbert):

Maths as a huge **game**

Goal: to prove theorems by following inference rules

- **Proof theory**: design & study of **rule systems** capturing maths

$$\frac{x \text{ is of type } A \quad f \text{ is a function from } A \text{ to } B}{f(x) \text{ is of type } B} \rightsquigarrow \frac{\vdash x : A \quad \vdash f : A \rightarrow B}{\vdash f(x) : B}$$

- *Generic patterns* of **computation** as symbolic **rules**
- Constructivist school (Brouwer-Heyting-Kolmogorov):

Maths as a huge **computation**

Goal: to *implement theorems* by building *programs*

- **Type theory**: design & study of **rule systems** capturing (sound) programming

Dream: a powerful language/software environment for both:

- formal **mathematics**
- verified **programming**

Dream: a **user-friendly**, yet powerful language/software environment for both:

- formal **mathematics**
- verified **programming**

Problem: current interfaces (and incoming ones based on LLMs) stuck in **textual** and **verbal** form

Commands on **Symbolic expressions**
Proofs Statements

The screenshot shows the Lean 4 IDE with the file `GrothendieckTopos.lean` open. The editor displays the following code:

```

131 noncomputable instance associatedFunctor_iso_sheafToPresheaf_obj_obj
132   (F : Sheaf J (Type w)) (X : Cop) :
133   ((associatedFunctor (yoneda » presheafToSheaf J (Type w))).obj F).obj X ≡
134   ((sheafToPresheaf J (Type w)).obj F).obj X where
135   hom := ((sheafificationAdjunction J (Type w)).homEquiv (yoneda.obj X.unop) F).to
136         yonedaEquiv.toFun
137   inv := yonedaEquiv.invFun »
138         ((sheafificationAdjunction J (Type w)).homEquiv (yoneda.obj X.unop) F).invFu
139   hom_inv_id := by tac constructs a term of the expected type by running the tactic(s) 'tac'.
140   inv_hom_id := by
141     set f := ((sheafificationAdjunction J (Type w)).homEquiv (yoneda.obj X.unop) F
142     set g := ((sheafificationAdjunction J (Type w)).homEquiv (yoneda.obj X.unop) F
143     let hom := yoneda.obj X.unop → (sheafToPresheaf J (Type w)).obj F
144     rw [← Category.assoc] Anthony Bordg, 2 months ago • add Grothendieck topo
145     apply Eq.trans (b := (yonedaEquiv.invFun » f » g) » yonedaEquiv.toFun)
146     case h₁ => rw [← Category.assoc]
147     case h₂ => have eq : f » g = 1 hom := by
148       apply Equiv.self_comp_symm
149       rw [eq]
150       rw [Category.comp_id]
151       apply Equiv.self_comp_symm
152
153 noncomputable def natTrans_associatedFunctor_sheafToPresheaf :
154   associatedFunctor (yoneda » presheafToSheaf J (Type w)) → sheafToPresheaf J (T
155   app F := { app := fun X => (associatedFunctor_iso_sheafToPresheaf_obj_obj J F X)
156             naturality := by
157               intros
158               apply funext
159               intro
160               unfold associatedFunctor_iso_sheafToPresheaf_obj_obj
161               simp only [Equiv.toFun_as_coe, types_comp_apply]
162               rw [yonedaEquiv_naturality']
163               simp only [EmbeddingLike.apply_eq_iff_eq]
164               apply Adjunction.homEquiv_naturality_left

```

The right sidebar shows the Lean Infview with the tactic state and goal 1goal:

```

▼ GrothendieckTopos.lean:144:4
▼ Tactic state
1goal
C : Type w
inst† : SmallCategory C
J : GrothendieckTopology C
F : Sheaf J (Type w)
X : Cop
f : (yoneda.obj (Opposite.unop X) → (sheafToPresheaf J (Type
w)).obj F) →
  ((presheafToSheaf J (Type w)).obj (yoneda.obj (Opposite.unop
X)) → F) := ((sheafificationAdjunction J (Type w)).homEquiv
(yoneda.obj (Opposite.unop X)) F).invFun
g : ((presheafToSheaf J (Type w)).obj (yoneda.obj (Opposite.unop
X)) → F) →
  (yoneda.obj (Opposite.unop X) → (sheafToPresheaf J (Type
w)).obj F) := ((sheafificationAdjunction J (Type w)).homEquiv
(yoneda.obj (Opposite.unop X)) F).toFun
hom : Type w := yoneda.obj (Opposite.unop X) → (sheafToPresheaf
J (Type w)).obj F
⊢ (yonedaEquiv.invFun » f) » g » yonedaEquiv.toFun = 1
  (((sheafToPresheaf J (Type w)).obj F).obj X)

```

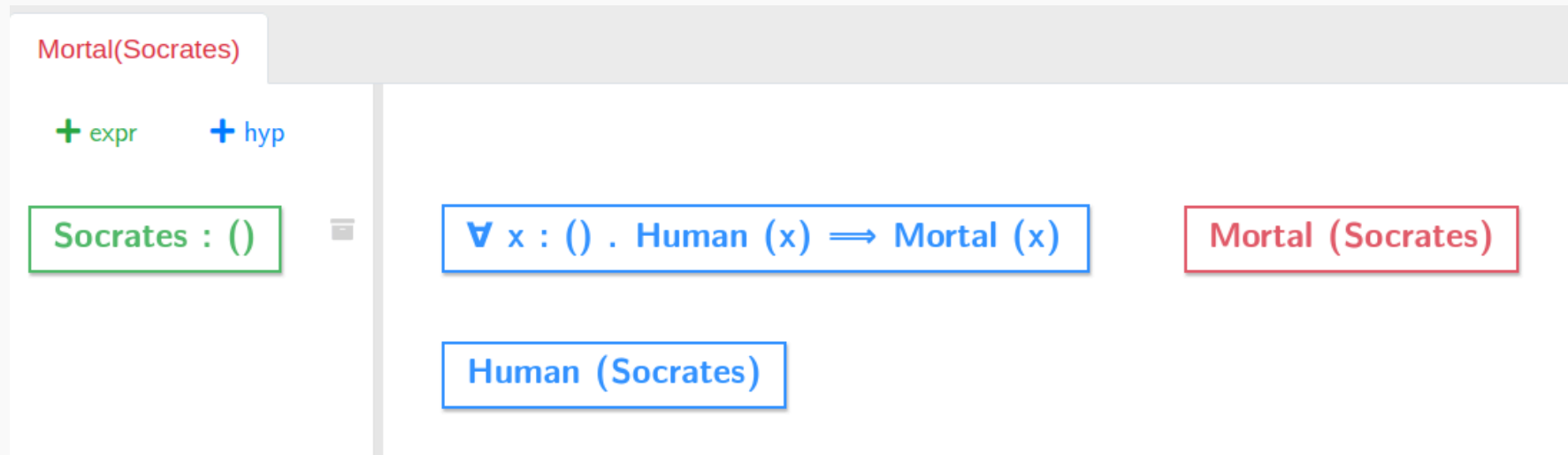
The bottom status bar shows the file is named `subobject-classifier` and the cursor is at line 144, column 5.

Proof-by-Action

Solution: **no-code** interface for proof assistants

↳ more **graphical** and **gestural** paradigm

Direct manipulation of **Formulas**
Proofs Statements

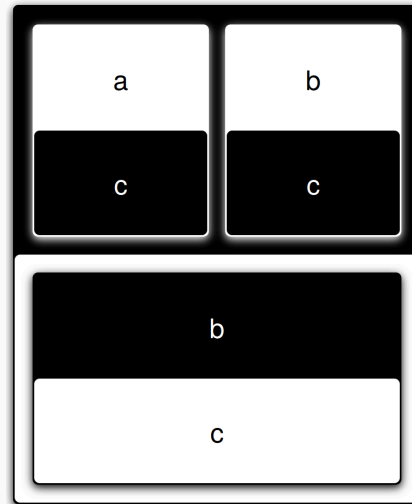


Proof-by-Action

Solution: **no-code** interface for proof assistants

↳ more graphical and **gestural** paradigm

Direct manipulation of Boxes
Proofs Statements

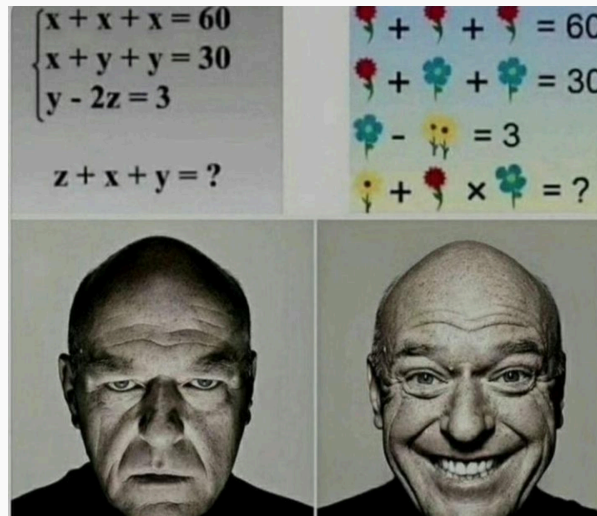


Proof-by-Action

Solution: **no-code** interface for proof assistants

↳ more graphical and **gestural** paradigm

Direct manipulation of Flowers 
Proofs Statements



Symbolic Manipulations

A demo is worth a thousand words!

Paradigm

- Fully graphical: **no textual** proof language
- Both **spatial** and **temporal**:

proof = **gesture sequence**

- **Different modes of reasoning with a single “syntax”:**

Technique	Action	Semantics	Proof theory
Proof-by-Pointing (Bertot, Kahn, and Théry 1994)	Click	Intro/Elim	Sequent calculus
Proof-by-Linking (Chaudhuri 2013)	Drag-and-Drop	Forward/Backward	Deep inference

Iconic Manipulations

Classical Logic: Existential Graphs

Three **diagrammatic** proof systems for **classical** logic:

- Alpha: *propositional* logic
- Beta: *first-order* logic
- Gamma: *higher-order* and *modal* logics

Three **diagrammatic** proof systems for **classical** logic:

- Alpha: *propositional* logic
- Beta: *first-order* logic
- Gamma: *higher-order* and *modal* logics

The three icons of Alpha

- Sheet of assertion
- Juxtaposition
- Cut

The three icons of Alpha

- Sheet of assertion

\vdash **true** (no assertion)

- Juxtaposition

- Cut

The three icons of Alpha

- Sheet of assertion

$$a \quad \vdash \quad \text{true (no assertion)}$$

- Juxtaposition

- Cut

The three icons of Alpha

- Sheet of assertion

$a \quad \vdash \quad \text{true (no assertion)}$
 $a \quad \vdash \quad a \text{ is true}$

- Juxtaposition

- Cut

The three icons of λ pha

- Sheet of assertion

\vdash **true** (no assertion)
 $a \vdash$ ***a* is true**

- Juxtaposition

$G \ H$

- Cut

The three icons of Λ pha

- Sheet of assertion

$\vdash \text{true}$ (no assertion)
 $a \vdash a \text{ is true}$

- Juxtaposition

$G \ H \vdash G \text{ is true and } H \text{ is true}$

- Cut

The three icons of Alpha

- Sheet of assertion

\vdash true (no assertion)
 $a \vdash a$ is true

- Juxtaposition

$G \ H \vdash G$ is true and H is true

- Cut



The three icons of Alpha

- Sheet of assertion

\vdash true (no assertion)
 $a \vdash a$ is true

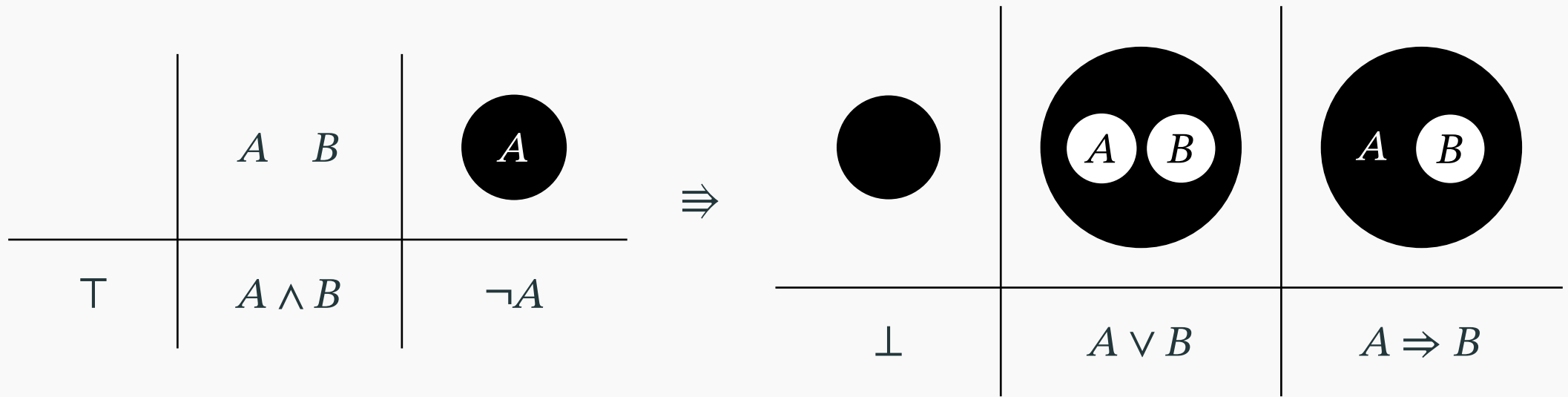
- Juxtaposition

$G \ H \vdash G$ is true and H is true

- Cut

$\neg G \vdash G$ is not true

Relationship with formulas



Illative transformations

Only 4 **edition** principles!

Illative transformations

Only 4 **edition** principles!

Iteration (copy-paste)			
G  \rightarrow G 			
G  \rightarrow G 			

Illative transformations

Only 4 **edition** principles!

Iteration (copy-paste)	Deiteration (unpaste)		
$G \quad \square \rightarrow G \quad \boxed{G}$	$G \quad \boxed{G} \rightarrow G \quad \blacksquare$		
$G \quad \blacksquare \rightarrow G \quad \square$	$G \quad \square \rightarrow G \quad \boxed{G}$		

Illative transformations

Only 4 **edition** principles!

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	
$G \quad \square \rightarrow G \quad \boxed{G}$	$G \quad \boxed{G} \rightarrow G \quad \square$	$\rightarrow G$	
$G \quad \blacksquare \rightarrow G \quad \boxed{G}$	$G \quad \boxed{G} \rightarrow G \quad \square$		

Illative transformations

Only 4 **edition** principles!

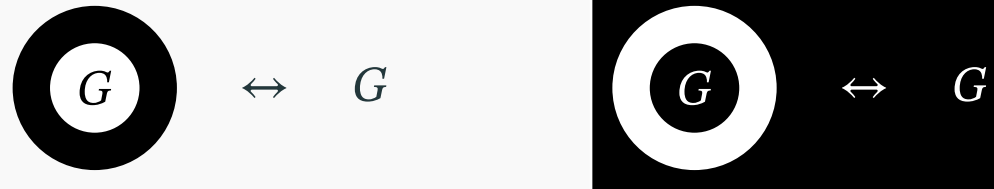
Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
$G \quad \square \rightarrow G \quad \boxed{G}$ $G \quad \blacksquare \rightarrow G \quad \boxed{G}$	$G \quad \boxed{G} \rightarrow G \quad \blacksquare$ $G \quad \boxed{G} \rightarrow G \quad \square$	$\rightarrow G$	$G \rightarrow$

Illative transformations

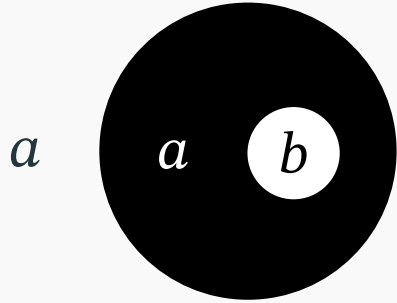
Only 4 **edition** principles!

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
$G \quad \square \rightarrow G \quad \boxed{G}$ $G \quad \blacksquare \rightarrow G \quad \boxed{G}$	$G \quad \boxed{G} \rightarrow G \quad \blacksquare$ $G \quad \boxed{G} \rightarrow G \quad \square$	$\blacksquare \rightarrow G$	$G \rightarrow \square$

and a **space** principle, the **Double-cut** law:



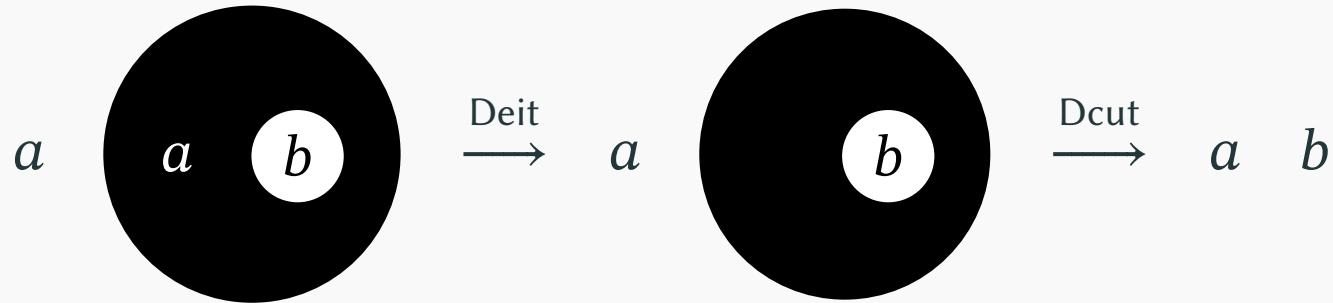
Example: modus ponens



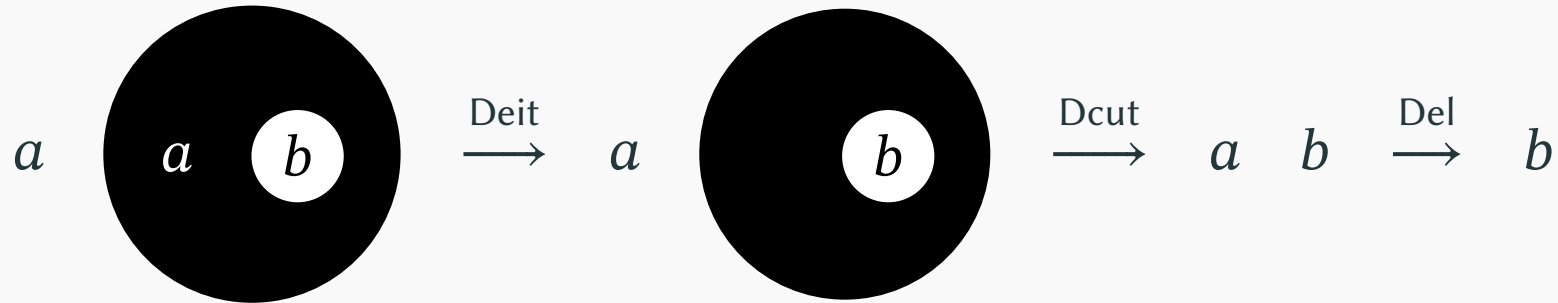
Example: modus ponens



Example: modus ponens

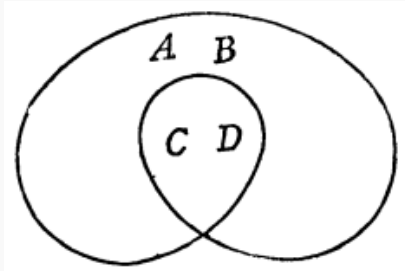


Example: modus ponens



Intuitionistic Logic: Flowers

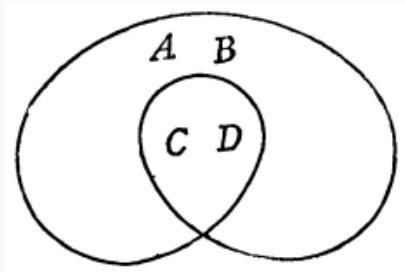
The scroll



I thought I ought to take the general form of argument as the basal form of composition of signs in my diagrammatization; and this necessarily took the form of a “scroll”, that is [...] a curved line without contrary flexure and returning into itself after once crossing itself.

— (Peirce 1906, pp. 533-534)

The scroll



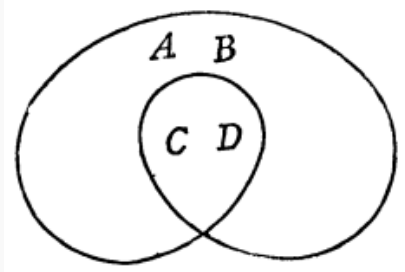
$$A \wedge B \Rightarrow C \wedge D$$

I thought I ought to take the general form of argument as the basal form of composition of signs in my diagrammatization; and this necessarily took the form of a “scroll”, that is [...] a curved line without contrary flexure and returning into itself after once crossing itself.

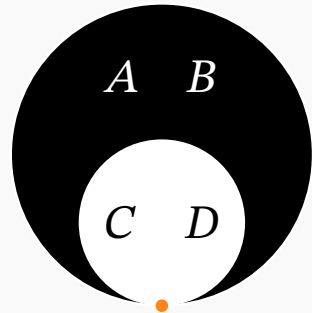
— (Peirce 1906, pp. 533-534)

- “conditional de inesse” = **classical** implication

The scroll



$$A \wedge B \Rightarrow C \wedge D$$



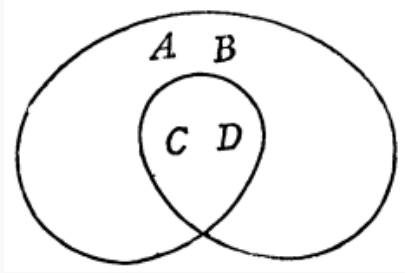
$$\neg(A \wedge B \wedge \neg(C \wedge D))$$

I thought I ought to take the general form of argument as the basal form of composition of signs in my diagrammatization; and this necessarily took the form of a “scroll”, that is [...] a curved line without contrary flexure and returning into itself after once crossing itself.

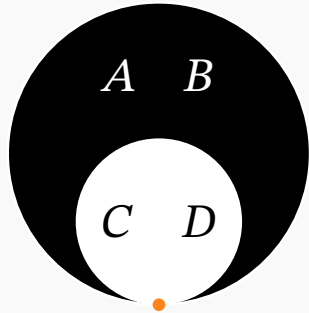
— (Peirce 1906, pp. 533-534)

- “conditional de inesse” = **classical** implication
- ↳ scroll = two **nested cuts**

The scroll



$$A \wedge B \Rightarrow C \wedge D$$

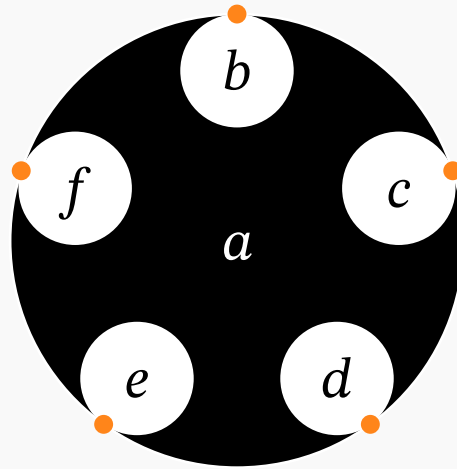


$$\neg(A \wedge B \wedge \neg(C \wedge D))$$

I thought I ought to take the general form of argument as the basal form of composition of signs in my diagrammatization; and this necessarily took the form of a “scroll”, that is [...] a curved line without contrary flexure and returning into itself after once crossing itself.

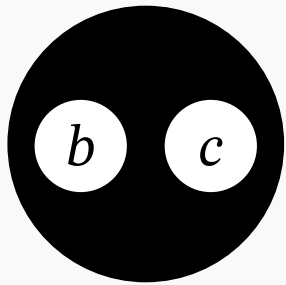
— (Peirce 1906, pp. 533-534)

- “conditional de inesse” = **classical** implication
↳ scroll = two **nested cuts**
- Peirce also introduced \Rightarrow in logic! (Lewis 1920, p. 79)

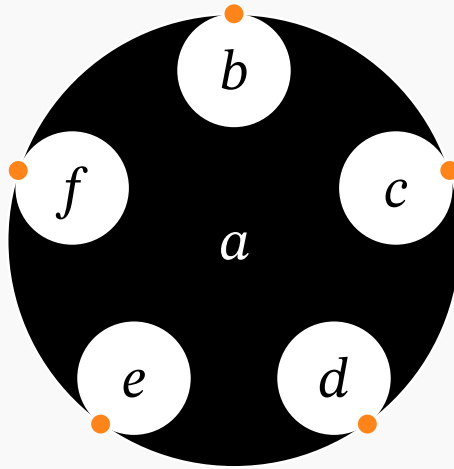


$$n = 5$$

Classical

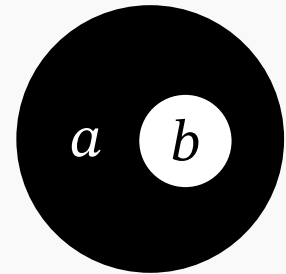


$$b \vee c$$



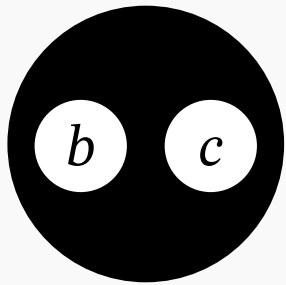
$$n = 5$$

Classical

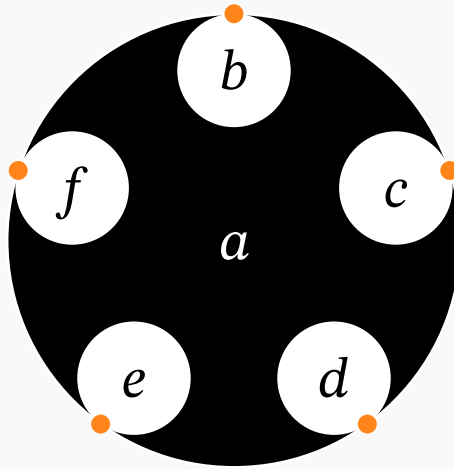


$$a \Rightarrow b$$

Classical



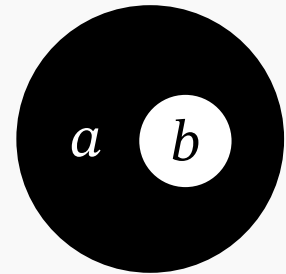
$$b \vee c$$



$$a \Rightarrow b \vee c \vee d \vee e \vee f$$

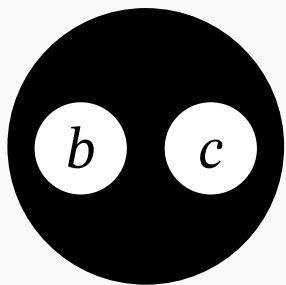
$$n = 5$$

Classical



$$a \Rightarrow b$$

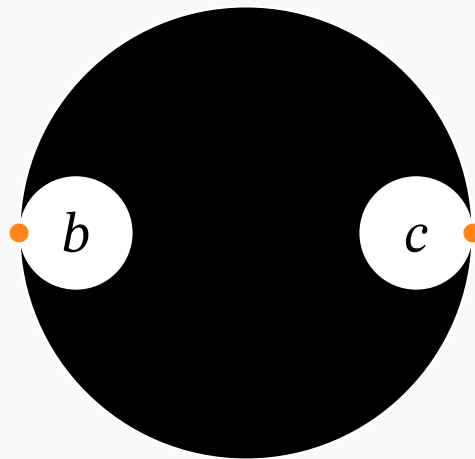
Intuitionistic



$$\neg(\neg b \wedge \neg c)$$

\neq

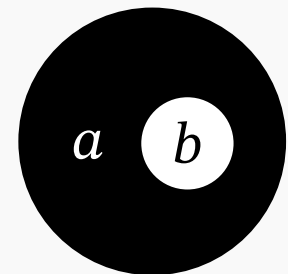
Continuity!



$$b \vee c$$

$$n = 2$$

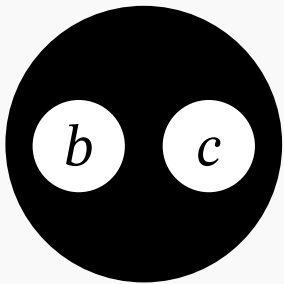
Classical



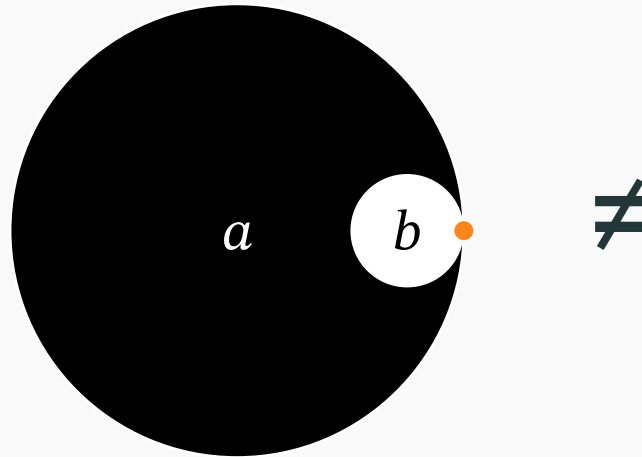
$$a \Rightarrow b$$

Continuity! Generalizes Peirce's scroll

Intuitionistic



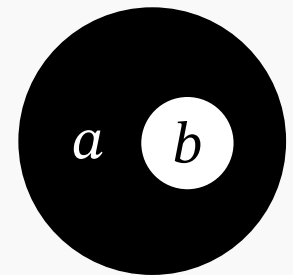
$$\neg(\neg b \wedge \neg c)$$



$$a \Rightarrow b$$

$$n = 1$$

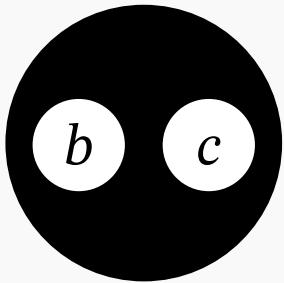
Intuitionistic



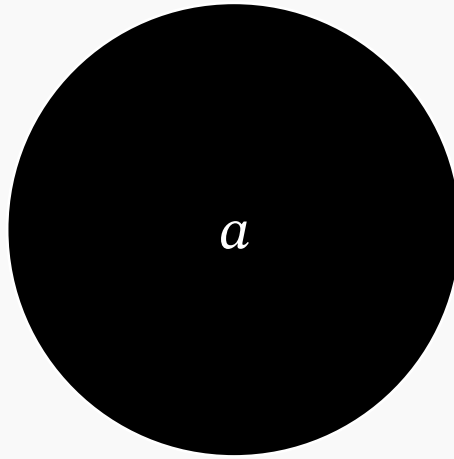
$$\neg(a \wedge \neg b)$$

Continuity! Generalizes Peirce's scroll and cut

Intuitionistic



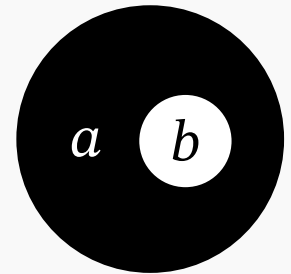
$$\neg(\neg b \wedge \neg c)$$



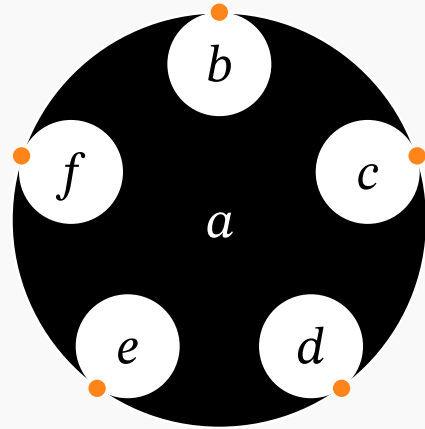
$$\neg a \triangleq a \Rightarrow \perp$$

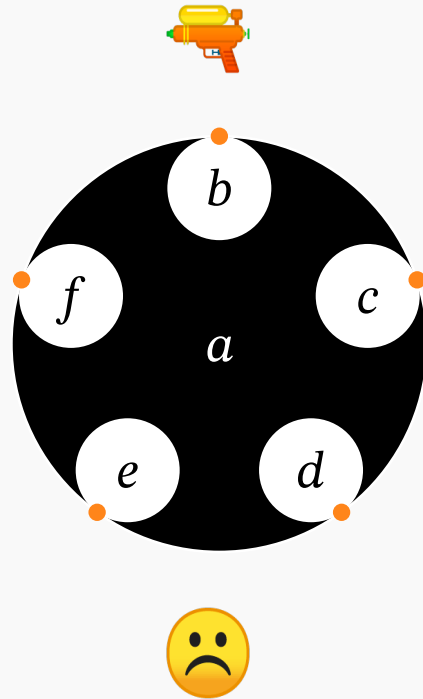
$$n = 0$$

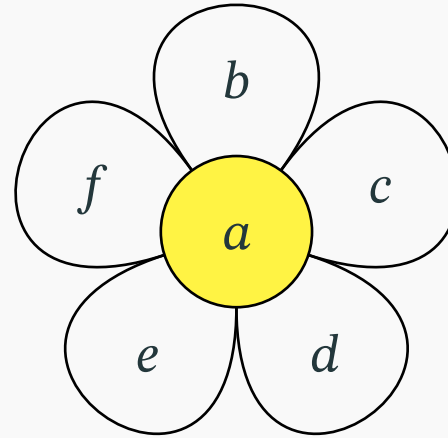
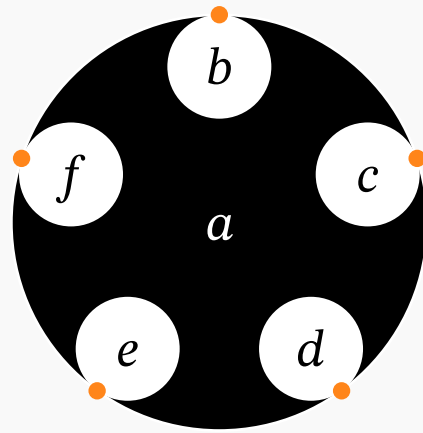
Intuitionistic



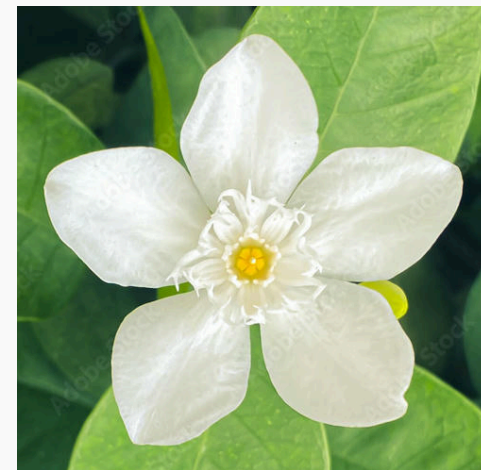
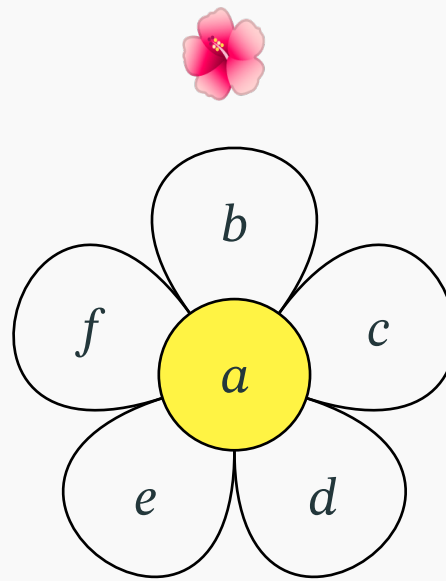
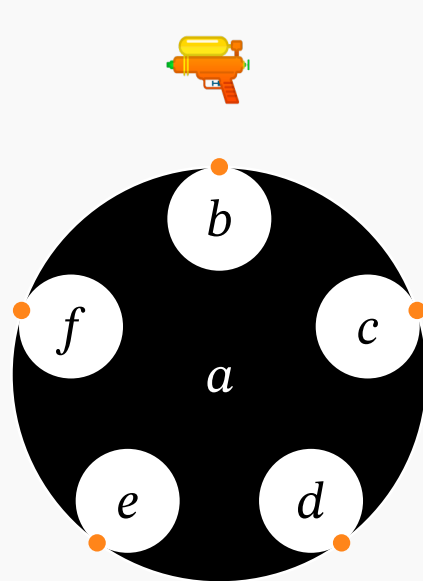
$$\neg(a \wedge \neg b)$$







Turn inloops into petals.



“Make love, not war”

Corollaries

The original “theorems” of geometry were those propositions that Euclid proved, while the **corollaries** were simple deductions from the theorems inserted by Euclid’s commentators and editors. They are said to have been marked the figure of a little garland (or **corolla**), in the origin.

— Peirce, MS 514 (1909) (Peirce 1976)

Corollaries

The original “theorems” of geometry were those propositions that Euclid proved, while the **corollaries** were simple deductions from the theorems inserted by Euclid’s commentators and editors. They are said to have been marked the figure of a little garland (or **corolla**), in the origin.

— Peirce, MS 514 (1909) (Peirce 1976)

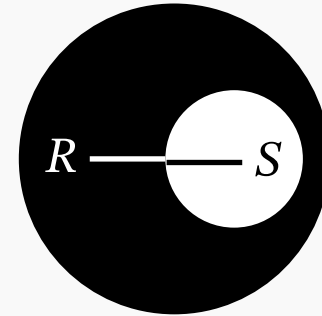
Petals = (possible) **corolla**-ries of pistil!

Predicate Logic: Gardens

Lines of Identity


In Beta, **quantifiers** and **variables** are represented with **lines**.

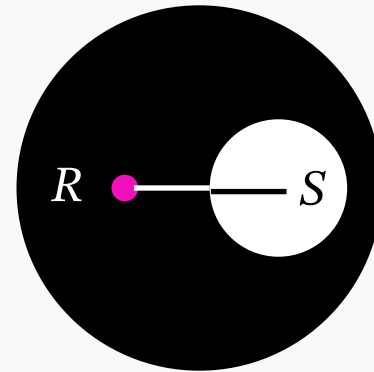
P — Q



Lines of Identity

In Beta, **quantifiers** and **variables** are represented with **lines**.

P   Q



quantifier location = **outermost point**

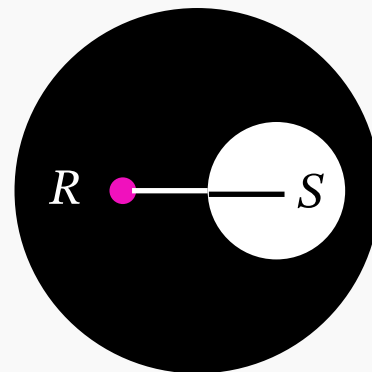
Lines of Identity

In Beta, **quantifiers** and **variables** are represented with **lines**.

P ● — Q

$\exists x.P(x) \wedge Q(x)$

existential graphs!




$\forall x.R(x) \Rightarrow S(x)$

$\cong \neg \exists x.R(x) \wedge \neg S(x)$

quantifier type = **outermost point polarity**

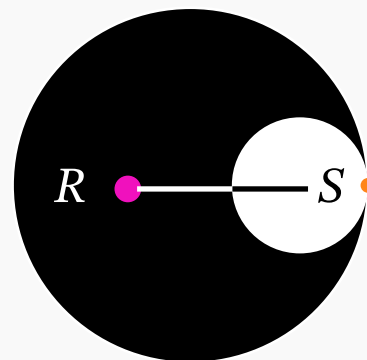
Lines of Identity

Problem: no De Morgan duality in intuitionistic logic

P  Q

$\exists x.P(x) \wedge Q(x)$

existential graphs!



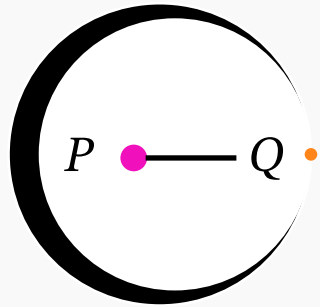
$\forall x.R(x) \Rightarrow S(x)$

$\not\equiv \neg \exists x.R(x) \wedge \neg S(x)$

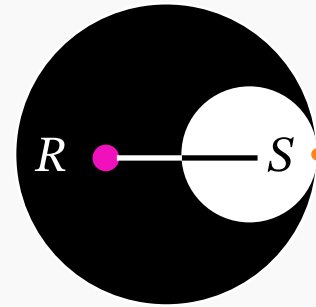
quantifier type = outermost point polarity

Intuitionistic quantification

Solution: polarity-invariant interpretation



$$\exists x.P(x) \wedge Q(x)$$

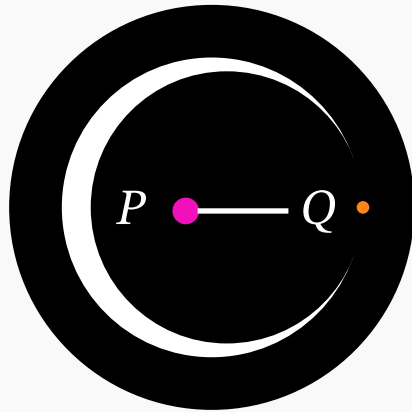


$$\forall x.R(x) \Rightarrow S(x)$$

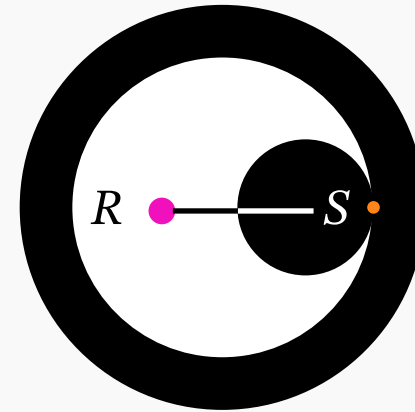
$\exists/\forall =$ inloop/outloop

Intuitionistic quantification

Solution: polarity-invariant interpretation



$$\neg(\exists x.P(x) \wedge Q(x))$$

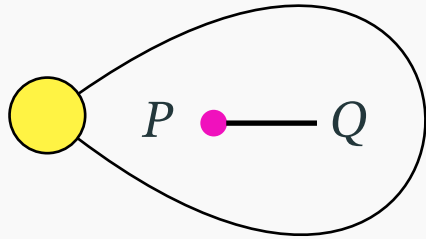


$$\neg(\forall x.R(x) \Rightarrow S(x))$$

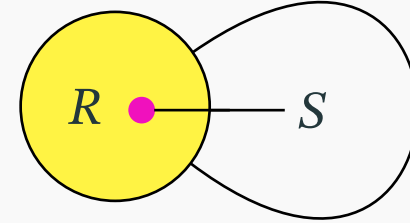
$\exists/\forall = \text{inloop/outloop}$

Intuitionistic quantification

Solution: polarity-invariant interpretation



$$\exists x. P(x) \wedge Q(x)$$

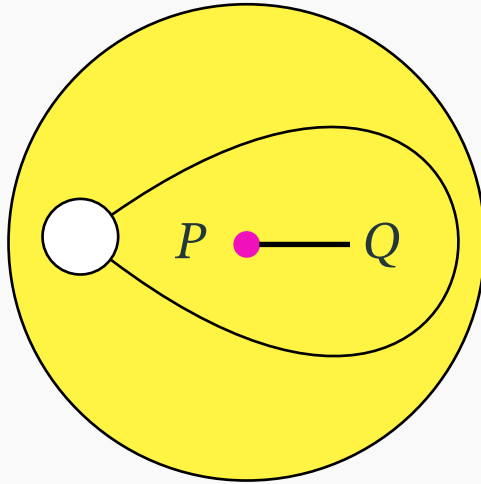


$$\forall x. R(x) \Rightarrow S(x)$$

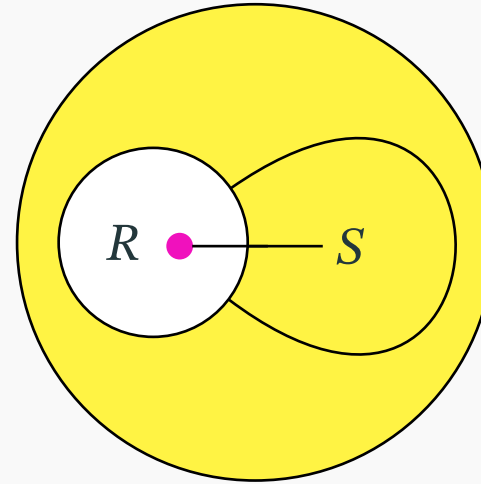
$\exists/\forall =$ petal/pistil

Intuitionistic quantification

Solution: polarity-invariant interpretation



$$\neg(\exists x.P(x) \wedge Q(x))$$

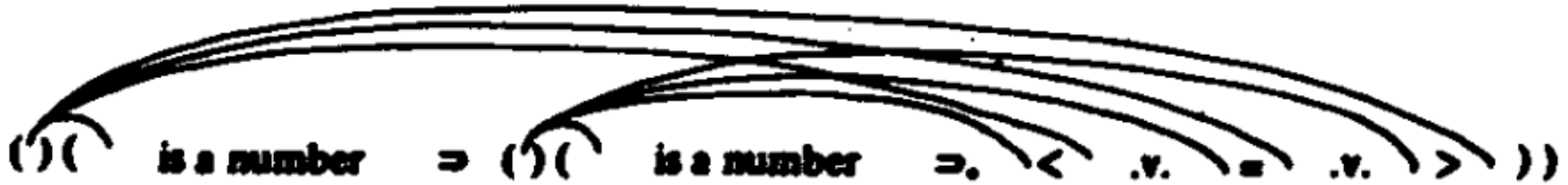


$$\neg(\forall x.R(x) \Rightarrow S(x))$$

$\exists/\forall =$ petal/pistil

Spaghetti statements

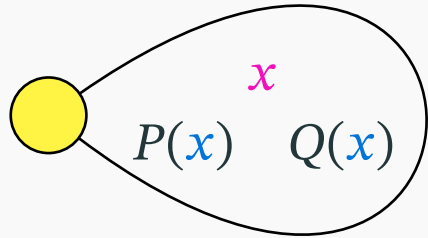
Problem: cables all over the place (well known in visual programming)



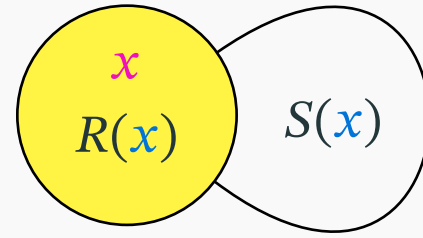
[These diagrams are] too cumbersome to recommend themselves as a practical notation.

— (Quine 1955, p. 70)

Solution: replace lines with good old **binders** and **variables**



$$\exists x. P(x) \wedge Q(x)$$



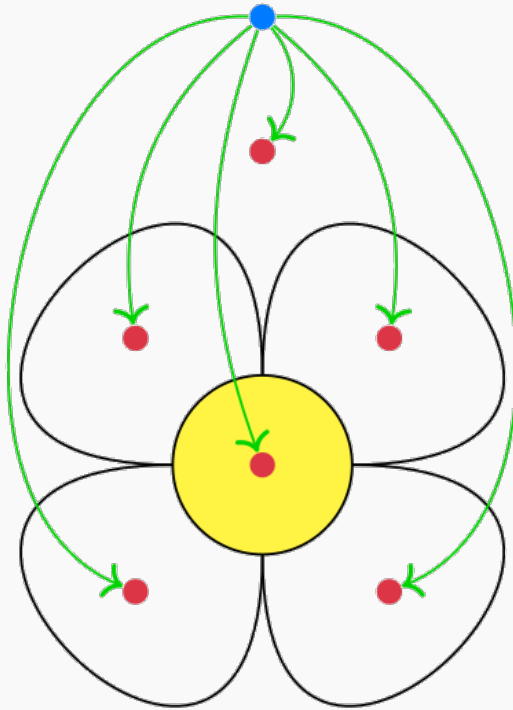
$$\forall x. R(x) \Rightarrow S(x)$$

garden = content of an area (binders + flowers)

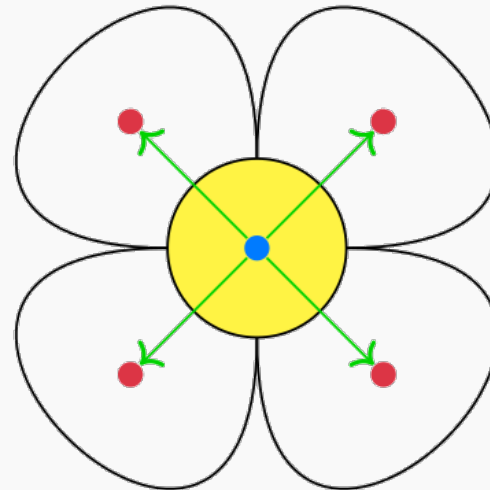
Reasoning with Flowers

Iteration and Deiteration

Justify a **target** by an **identical source**



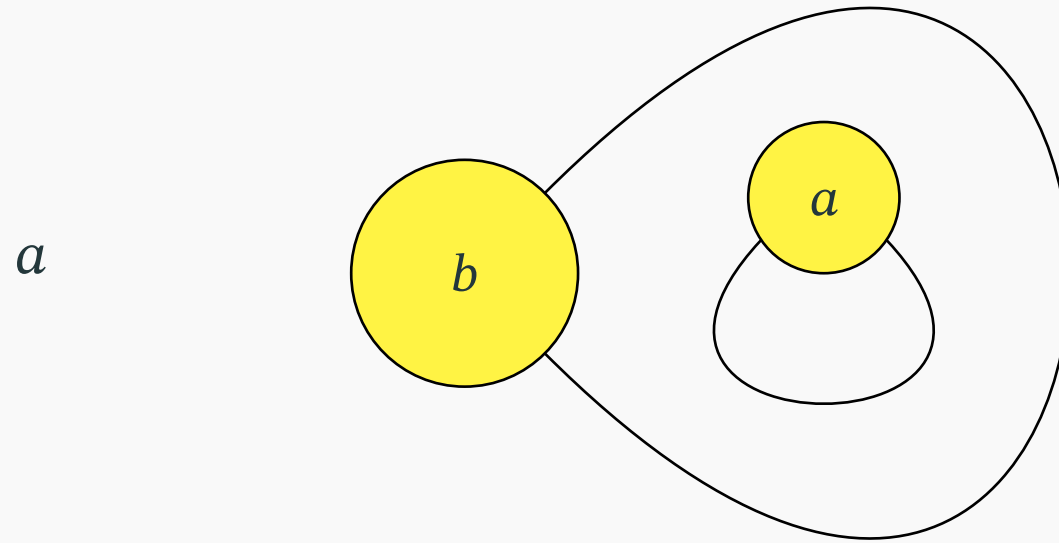
cross-pollination



self-pollination

Iteration and Deiteration

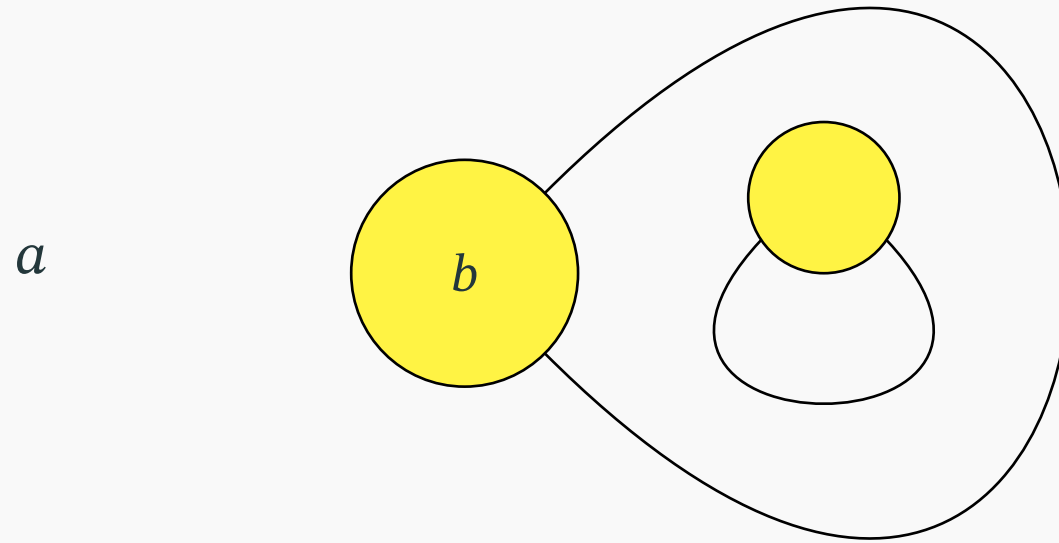
Works at arbitrary **depth**!



Cross-pollination

Iteration and Deiteration

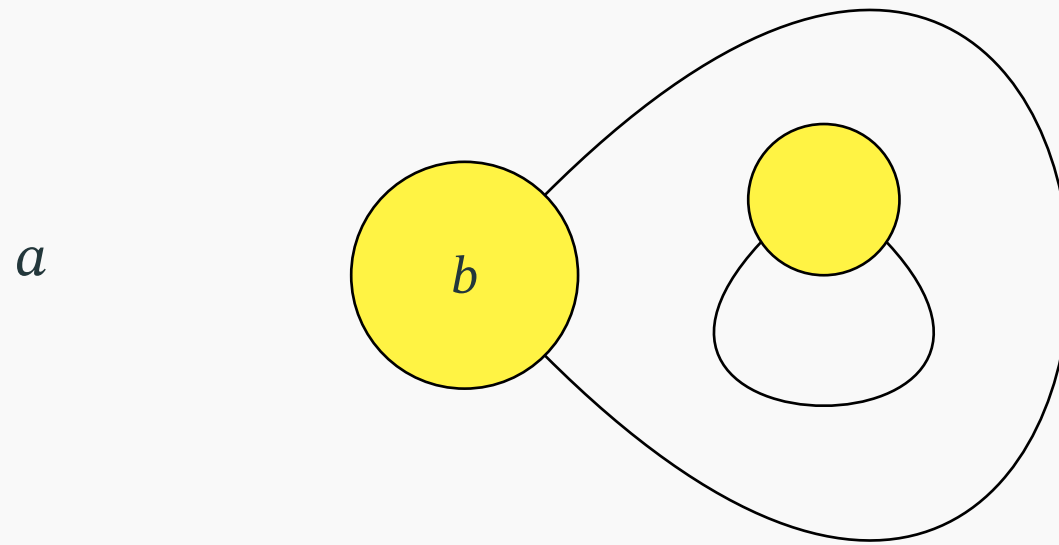
Works at arbitrary **depth**!



Cross-pollination

Iteration and Deiteration

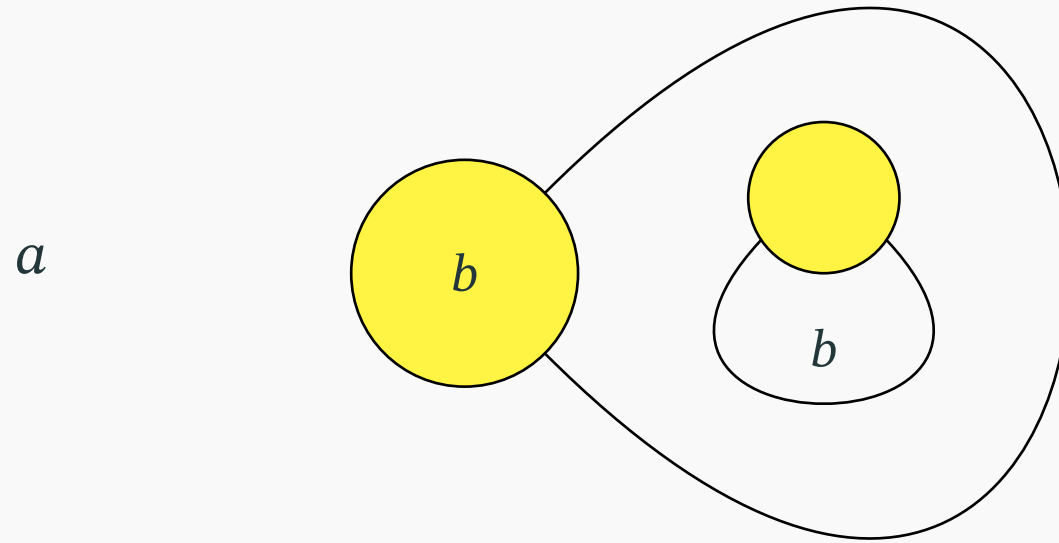
Works at arbitrary **depth**!



Self-pollination

Iteration and Deiteration

Works at arbitrary **depth**!



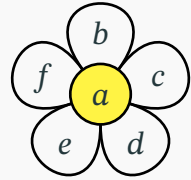
Self-pollination

Insertion and Deletion

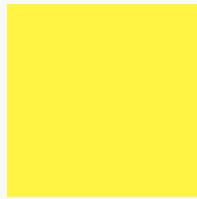
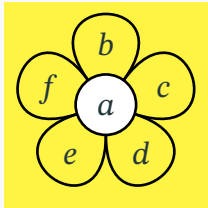
Split in two:

Flower

grow
→

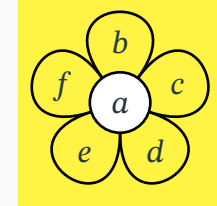
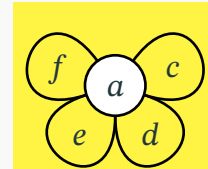


crop
→

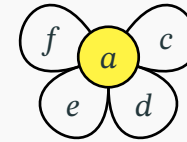
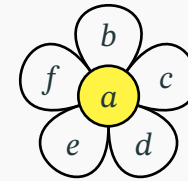


Petal

glue
→

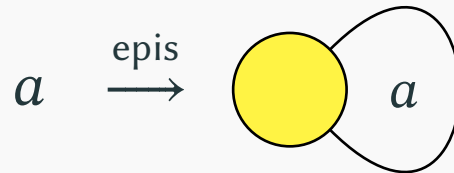


pull
→

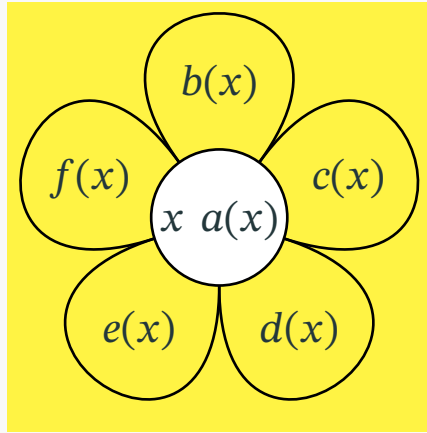


Backward reading: conclusion → premiss

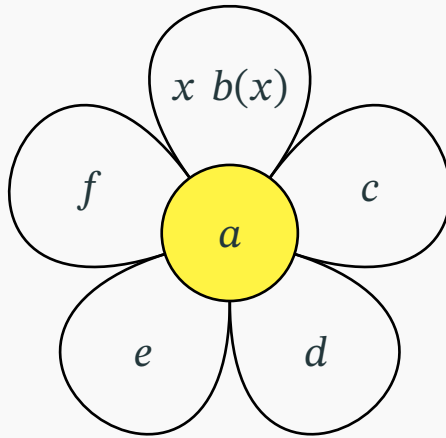
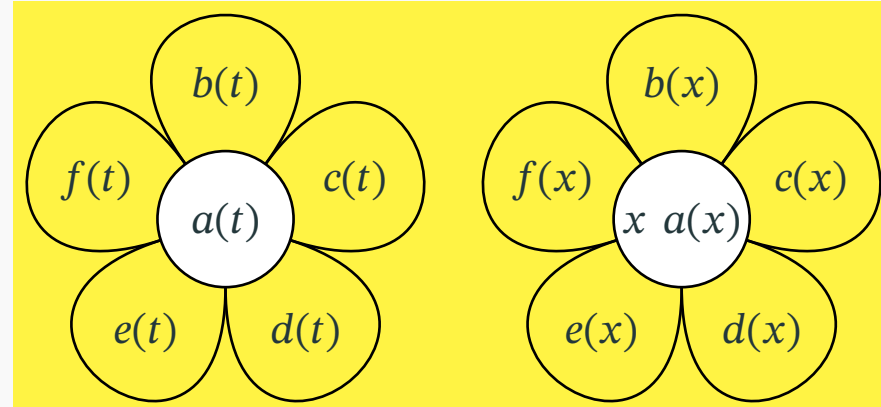
Intuitionistic restriction of **double-cut** principle:



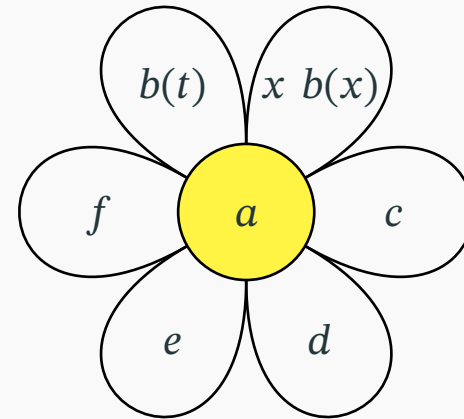
Instantiation



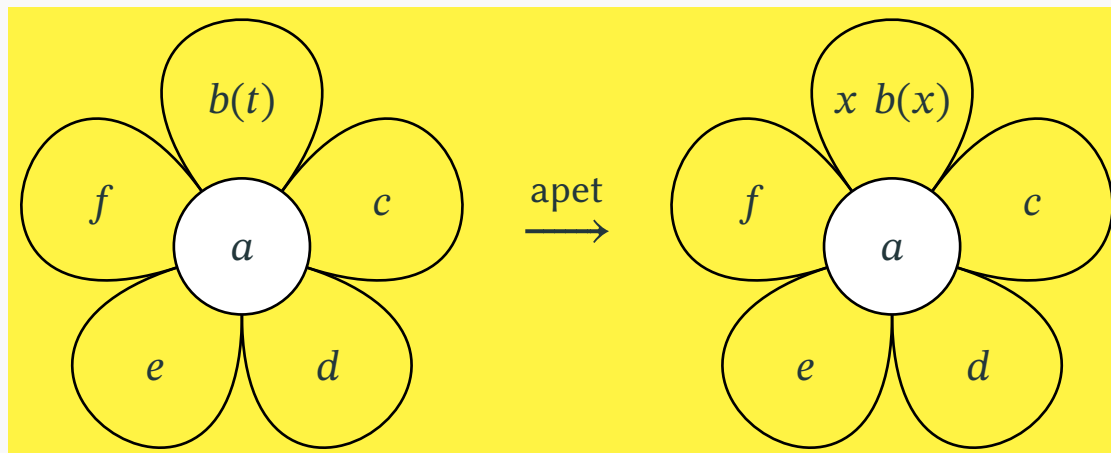
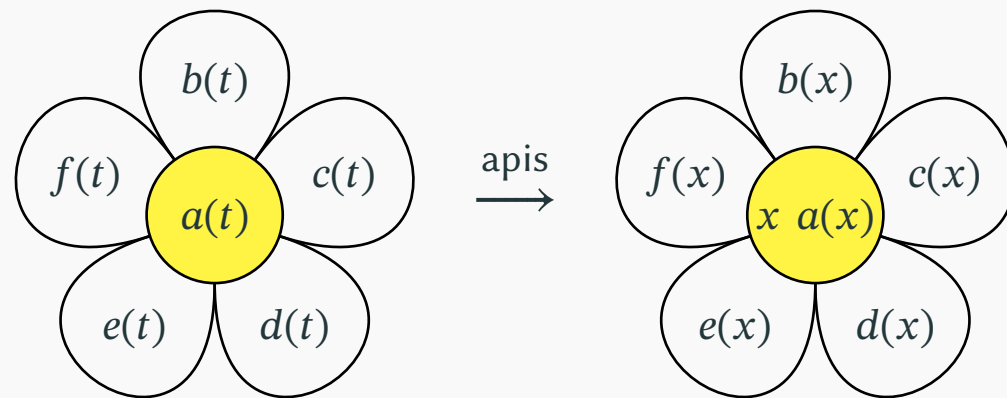
ipis
→



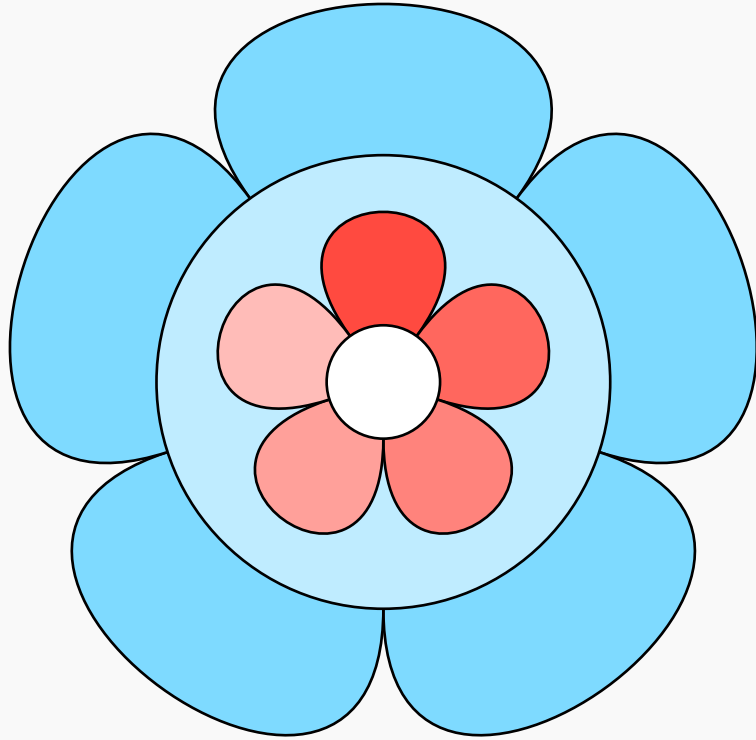
ipet
→



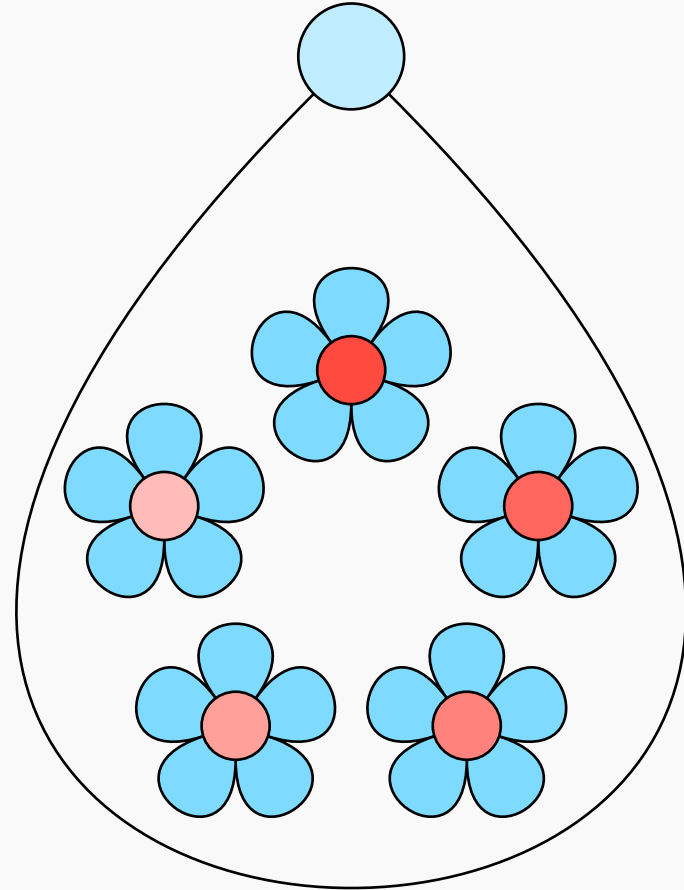
Abstraction



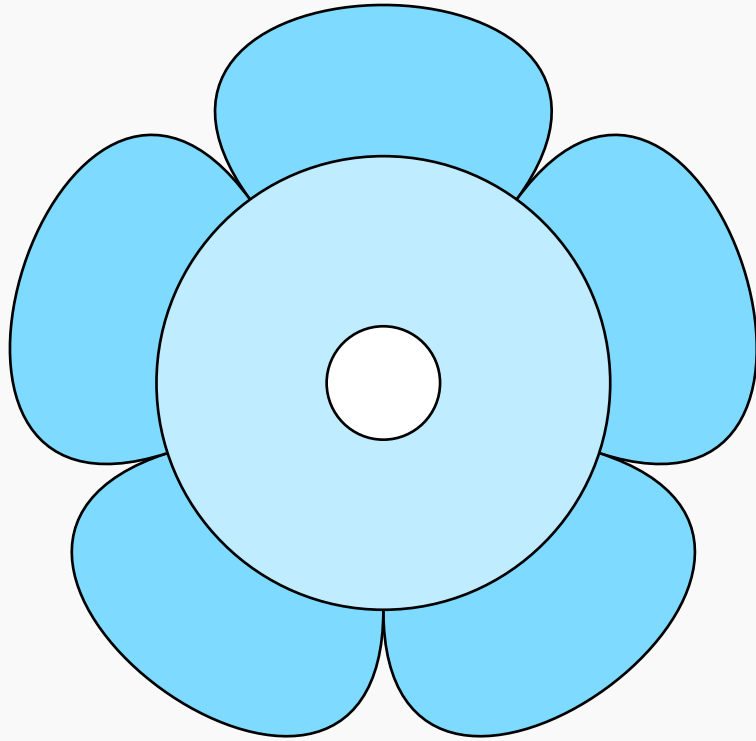
Case reasoning



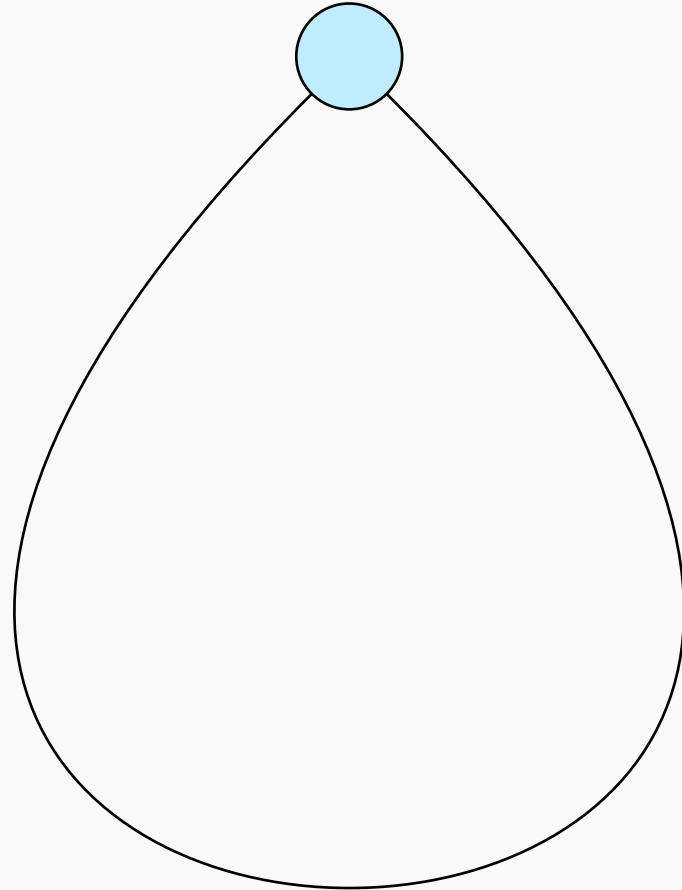
srep
→

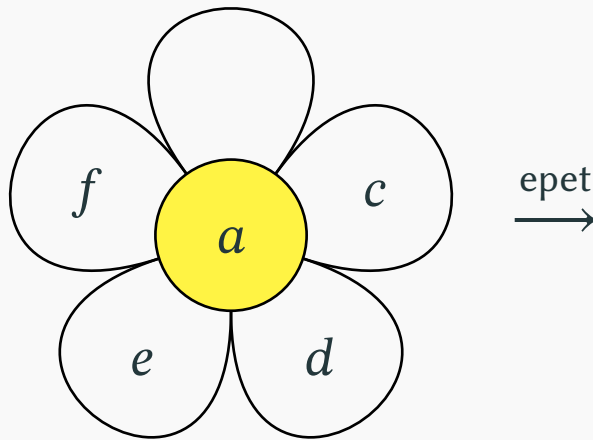


Ex falso quodlibet



srep
→





Metatheory: Nature vs. Culture

Natural rules ☼

$$\begin{array}{ccccccccc} \text{☼} & = & \text{(De)iteration} & \cup & \text{Instantiation} & \cup & \text{Scrolling} & \cup & \text{QED} & \cup & \text{Case reasoning} \\ & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} \\ & & \{\text{poll}\downarrow, \text{poll}\uparrow\} & & \{\text{ipis}, \text{ipet}\} & & \{\text{epis}\} & & \{\text{epet}\} & & \{\text{srep}\} \end{array}$$

Natural rules ☼

$$\begin{array}{ccccccccc} \text{☼} & = & \text{(De)iteration} & \cup & \text{Instantiation} & \cup & \text{Scrolling} & \cup & \text{QED} & \cup & \text{Case reasoning} \\ & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} \\ & & \{\text{poll}\downarrow, \text{poll}\uparrow\} & & \{\text{ipis}, \text{ipet}\} & & \{\text{epis}\} & & \{\text{epet}\} & & \{\text{srep}\} \end{array}$$

All rules are:

- **Invertible:** if $\Phi \longrightarrow \Psi$ then Ψ equivalent to Φ

Natural rules \clubsuit

$$\clubsuit = \underbrace{(\text{De})\text{iteration}}_{\{\text{poll}\downarrow, \text{poll}\uparrow\}} \cup \underbrace{\text{Instantiation}}_{\{\text{ipis}, \text{ipet}\}} \cup \underbrace{\text{Scrolling}}_{\{\text{epis}\}} \cup \underbrace{\text{QED}}_{\{\text{epet}\}} \cup \underbrace{\text{Case reasoning}}_{\{\text{srep}\}}$$

All rules are:

- **Invertible:** if $\Phi \longrightarrow \Psi$ then Ψ equivalent to Φ
 \hookrightarrow “Equational” reasoning

Natural rules ☼

$$\begin{array}{ccccccccc} \text{☼} & = & \text{(De)iteration} & \cup & \text{Instantiation} & \cup & \text{Scrolling} & \cup & \text{QED} & \cup & \text{Case reasoning} \\ & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} \\ & & \{\text{poll}\downarrow, \text{poll}\uparrow\} & & \{\text{ipis}, \text{ipet}\} & & \{\text{epis}\} & & \{\text{epet}\} & & \{\text{srep}\} \end{array}$$

All rules are:

- **Invertible:** if $\Phi \longrightarrow \Psi$ then Ψ equivalent to Φ
 \hookrightarrow “Equational” reasoning
- **Analytic:** if $\Phi \longrightarrow \Psi$ and a occurs in Ψ then a occurs in Φ

Natural rules ☼

$$\begin{array}{ccccccccc} \text{☼} & = & \text{(De)iteration} & \cup & \text{Instantiation} & \cup & \text{Scrolling} & \cup & \text{QED} & \cup & \text{Case reasoning} \\ & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} & & \underbrace{\hspace{1.5cm}} \\ & & \{\text{poll}\downarrow, \text{poll}\uparrow\} & & \{\text{ipis}, \text{ipet}\} & & \{\text{epis}\} & & \{\text{epet}\} & & \{\text{srep}\} \end{array}$$

All rules are:

- **Invertible:** if $\Phi \longrightarrow \Psi$ then Ψ equivalent to Φ
 - ↳ “Equational” reasoning
- **Analytic:** if $\Phi \longrightarrow \Psi$ and a occurs in Ψ then a occurs in Φ
 - ↳ Reduces proof-search space

$$\text{✂} = \underbrace{\text{Insertion}}_{\{\text{grow, glue}\}} \cup \underbrace{\text{Deletion}}_{\{\text{crop, pull}\}} \cup \underbrace{\text{Abstraction}}_{\{\text{apis, apet}\}}$$

$$\text{✂} = \underbrace{\text{Insertion}}_{\{\text{grow, glue}\}} \cup \underbrace{\text{Deletion}}_{\{\text{crop, pull}\}} \cup \underbrace{\text{Abstraction}}_{\{\text{apis, apet}\}}$$

- All rules are **non-invertible**
- Some rules are **non-analytic**

Theorem (Soundness): If $\Phi \vdash \Psi$ then $\Phi \vDash^{\mathcal{K}} \Psi$ in every Kripke structure \mathcal{K} .

Theorem (Soundness): If $\Phi \vdash \Psi$ then $\Phi \models^{\mathcal{K}} \Psi$ in every Kripke structure \mathcal{K} .

Theorem (Completeness): If $\Phi \models^{\mathcal{K}} \Psi$ in every Kripke structure \mathcal{K} , then $\Phi \vdash \Psi$.

Cut-elimination

Theorem (Soundness): If $\Phi \vdash \Psi$ then $\Phi \models^{\mathcal{K}} \Psi$ in every Kripke structure \mathcal{K} .

Theorem (Completeness): If $\Phi \models^{\mathcal{K}} \Psi$ in every Kripke structure \mathcal{K} , then $\Phi \vdash^* \Psi$.

Corollary (Admissibility of \multimap): If $\Phi \vdash \Psi$ then $\Phi \vdash^* \Psi$.

Cut-elimination

Theorem (Soundness): If $\Phi \vdash \Psi$ then $\Phi \models^{\mathcal{K}} \Psi$ in every Kripke structure \mathcal{K} .

Theorem (Completeness): If $\Phi \models^{\mathcal{K}} \Psi$ in every Kripke structure \mathcal{K} , then $\Phi \vdash^* \Psi$.

Corollary (Admissibility of \multimap): If $\Phi \vdash \Psi$ then $\Phi \vdash^* \Psi$.

Completeness of **analytic** fragment \multimap !

\hookrightarrow **normal form** for proofs

The Flower Prover

A demo is worth a thousand pictures!

GUI in the Proof-by-Action paradigm based on the flower calculus

- Represent flowers as nested **boxes**
- **Modal interface** to interpret gestural actions:

Proof mode \iff **Natural** (invertible and analytic) rules

Edit mode \iff **Cultural** (non-invertible) rules

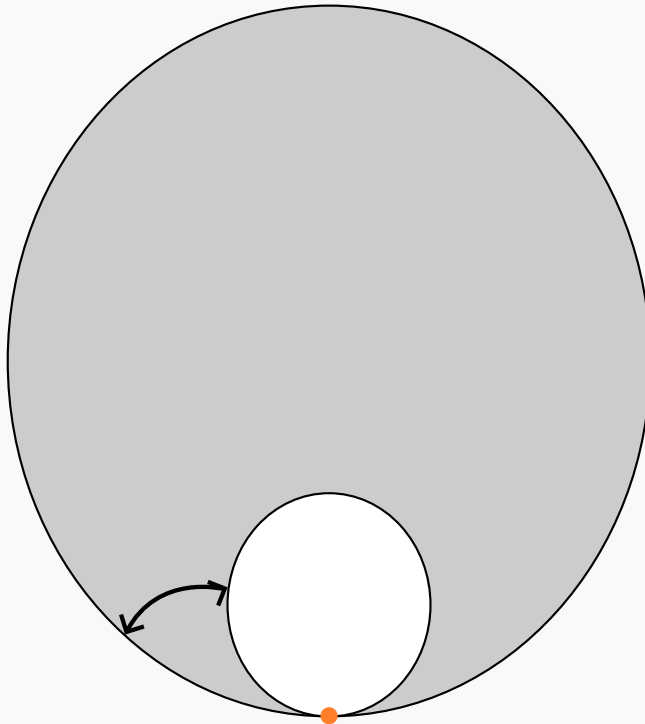
Navigation mode \iff **Contextual** closure (functoriality)

Towards Curry-Howard

Idea: **record** every move with **arrows**

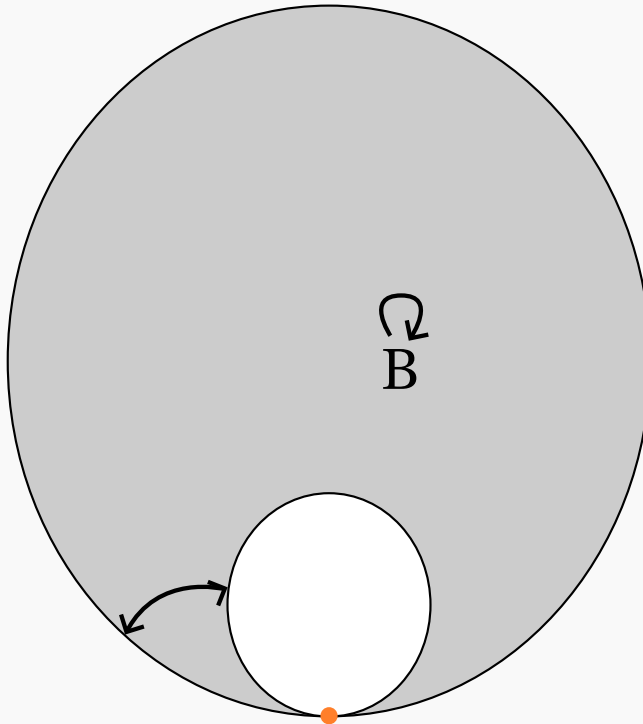
Scroll nets

Idea: **record** every move with arrows



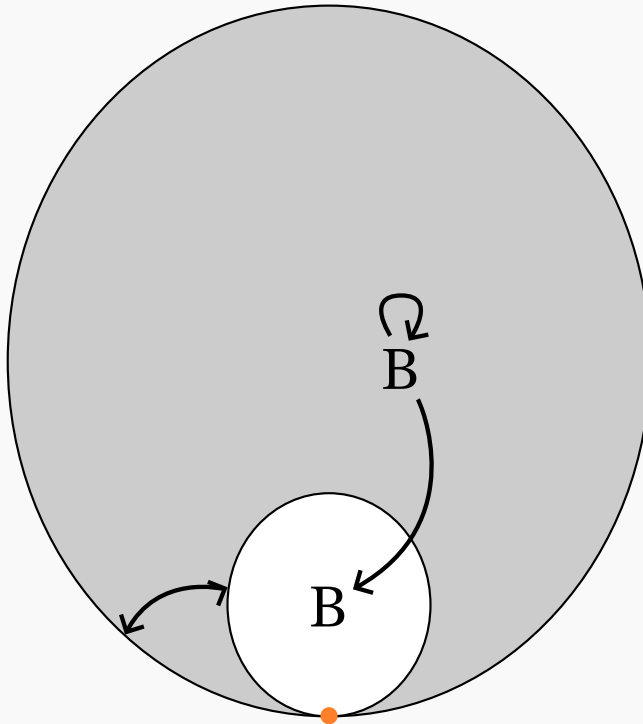
Scroll nets

Idea: **record** every move with arrows



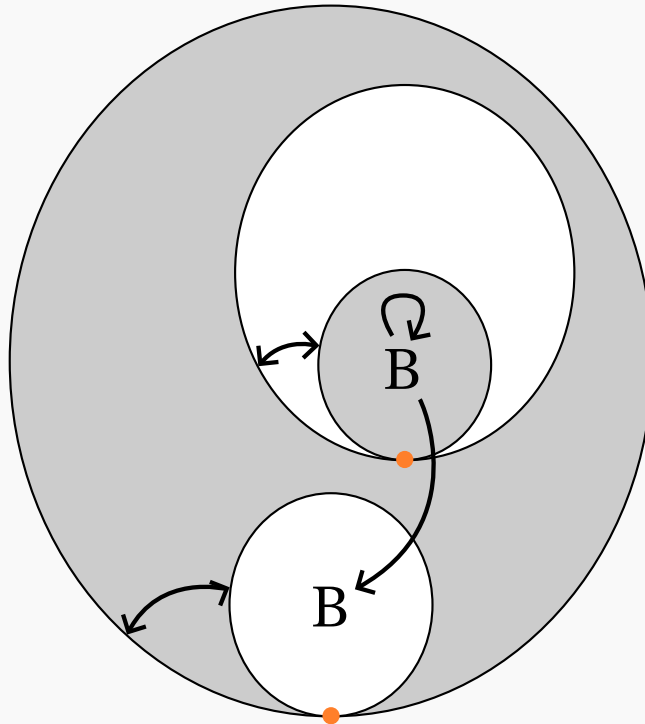
Scroll nets

Idea: **record** every move with arrows



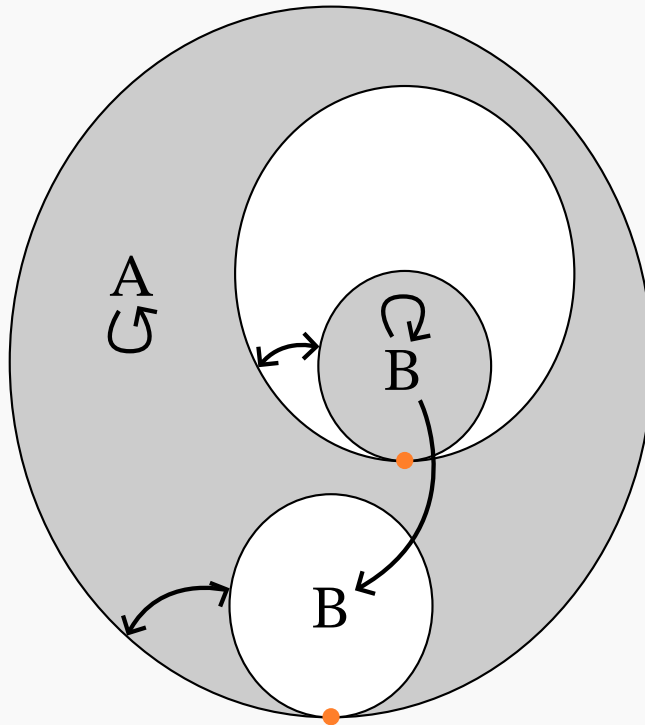
Scroll nets

Idea: **record** every move with arrows



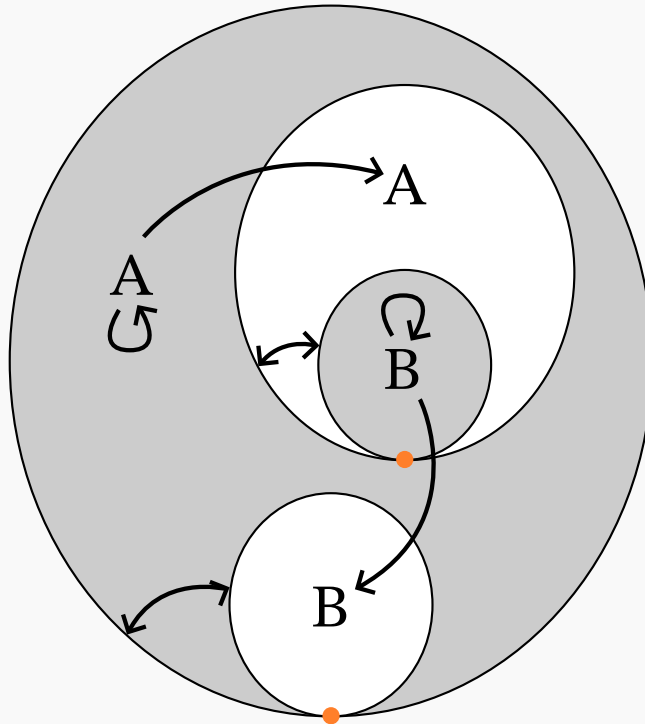
Scroll nets

Idea: **record** every move with arrows



Scroll nets



Idea: **record** every move with arrows



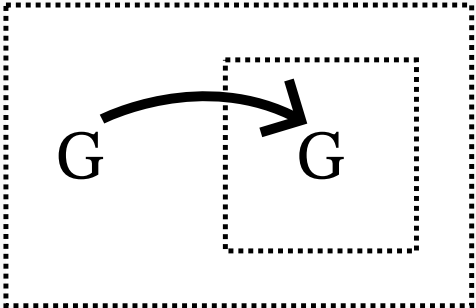
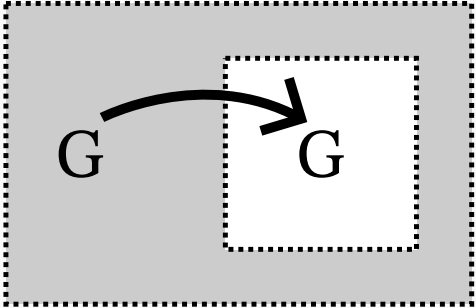
Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion

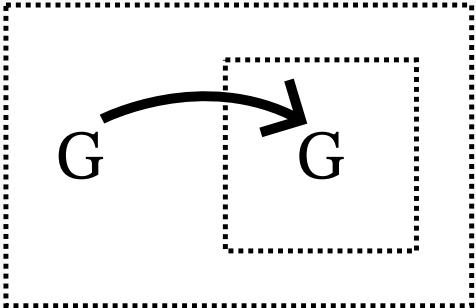
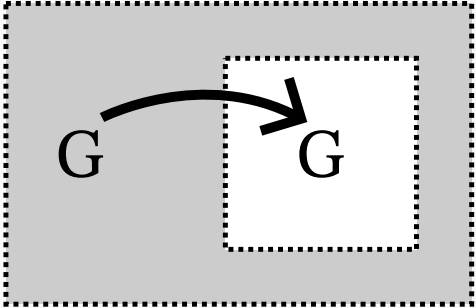
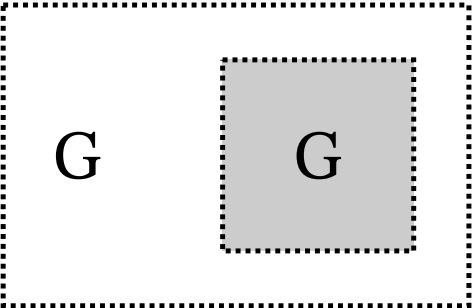
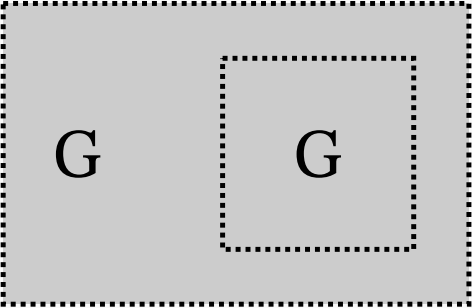
Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
			
			

Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
 			

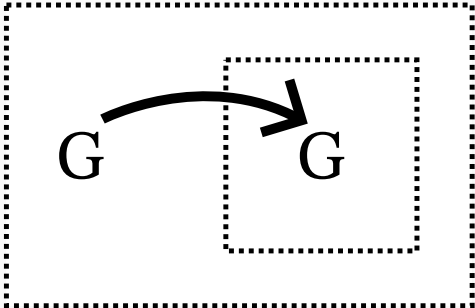
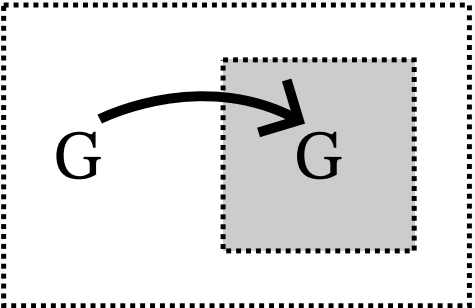
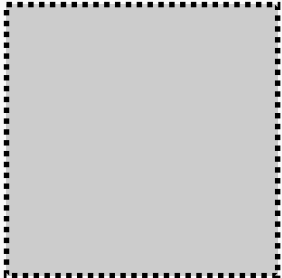
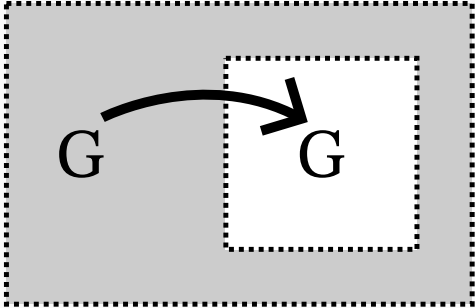
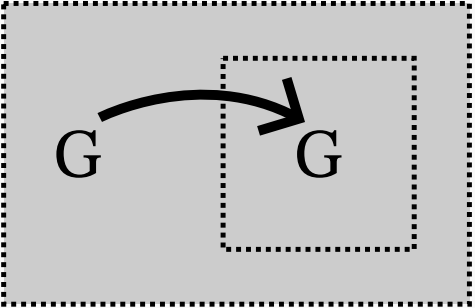
Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
 	 		

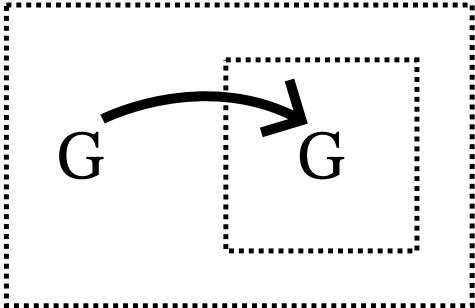
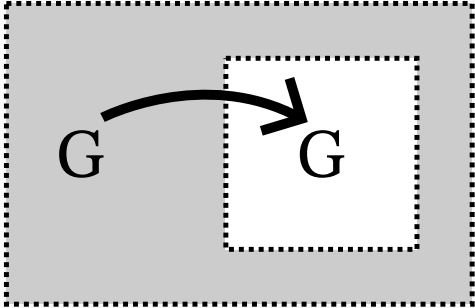
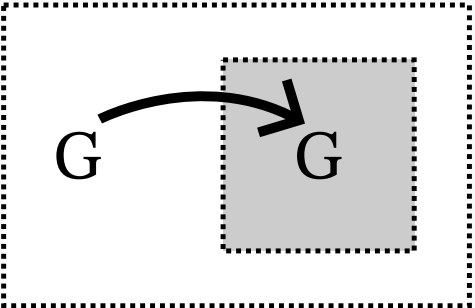
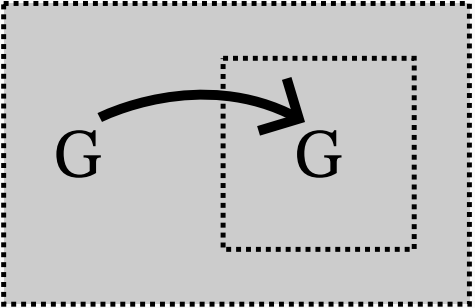
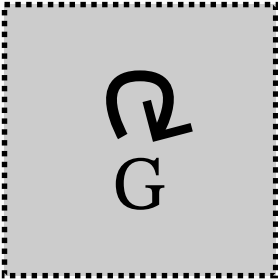
Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
<p>The diagram shows a large dashed rectangle containing a 'G' on the left and a smaller dashed rectangle on the right containing another 'G'. A curved arrow points from the left 'G' to the right 'G'. The entire diagram is enclosed in a solid border.</p>	<p>The diagram is identical to the Iteration diagram, but the smaller dashed rectangle on the right is shaded gray.</p>		
<p>The diagram is identical to the Iteration diagram, but the larger dashed rectangle is shaded gray.</p>	<p>The diagram is identical to the Deiteration diagram, but the larger dashed rectangle is shaded gray.</p>		

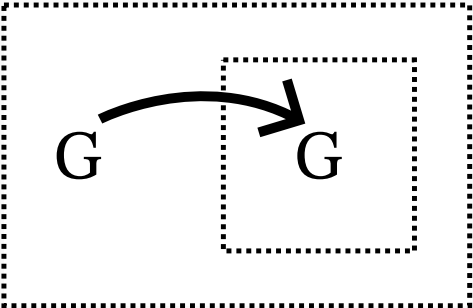
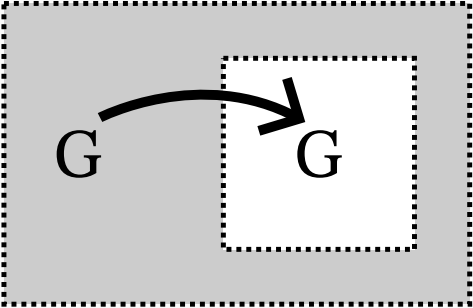
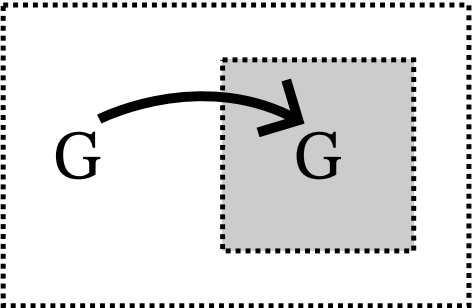
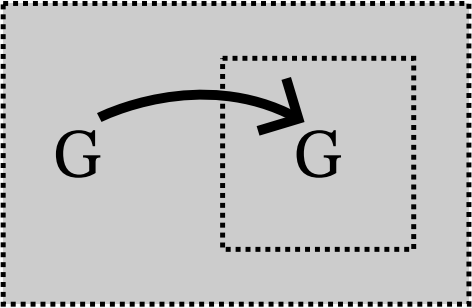
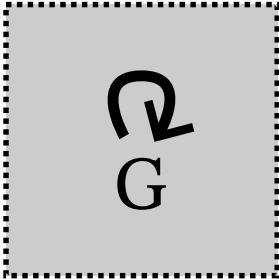
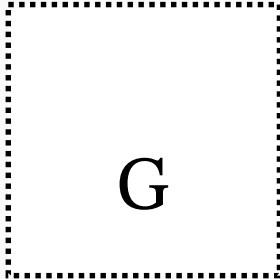
Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
			
			

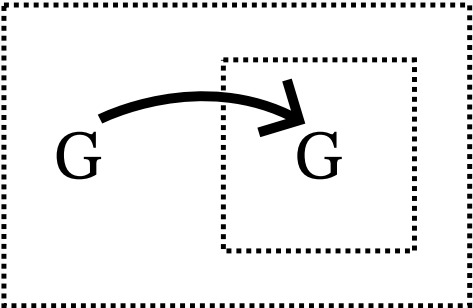
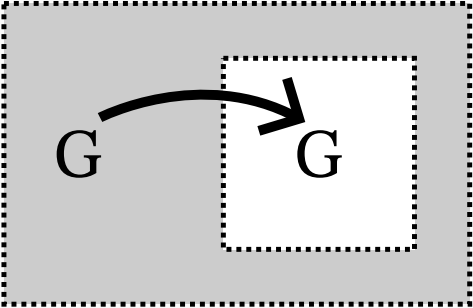
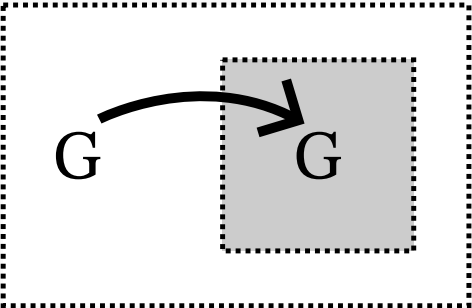
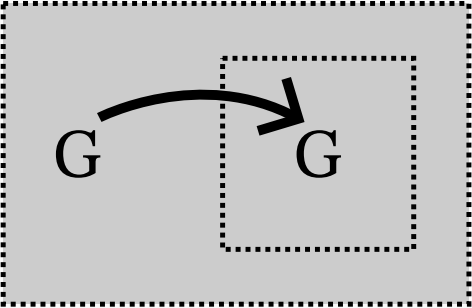
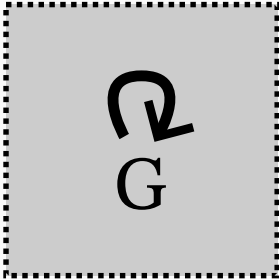
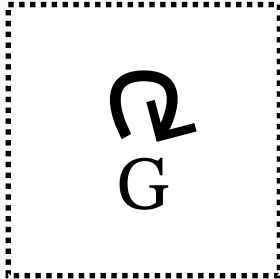
Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
 	 		

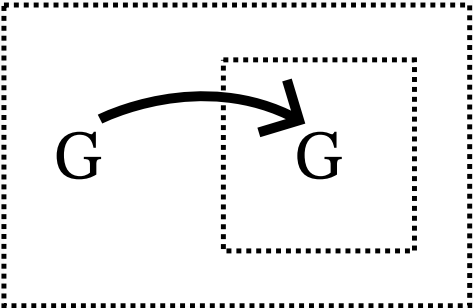
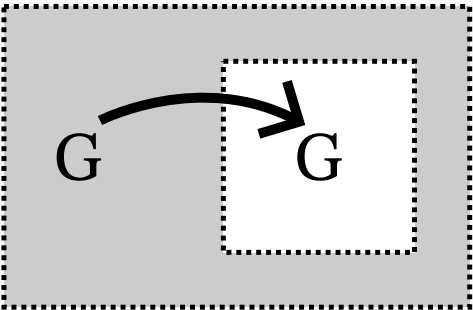
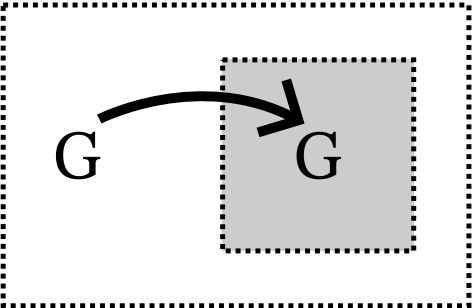
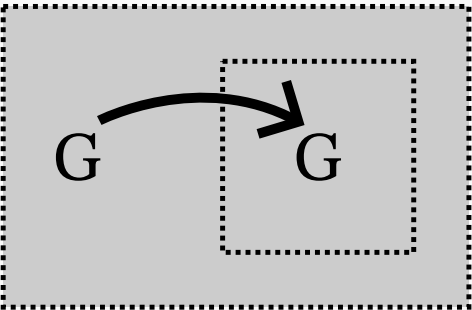
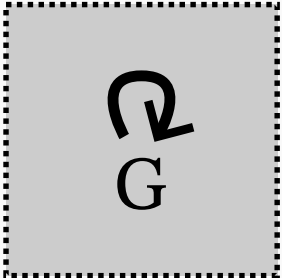
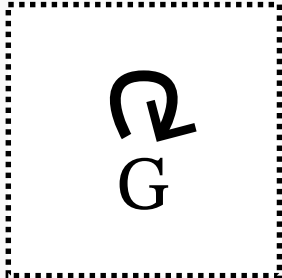
Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
 	 		

Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
 	 		

Justifications

Iteration (copy-paste)	Deiteration (unpaste)	Insertion	Deletion
 	 		

Structure of **digraph**, transformations determined by **polarity**

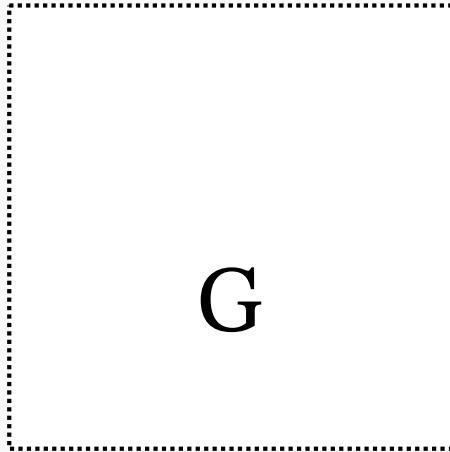
Interactions

Opening

Closing

Interactions

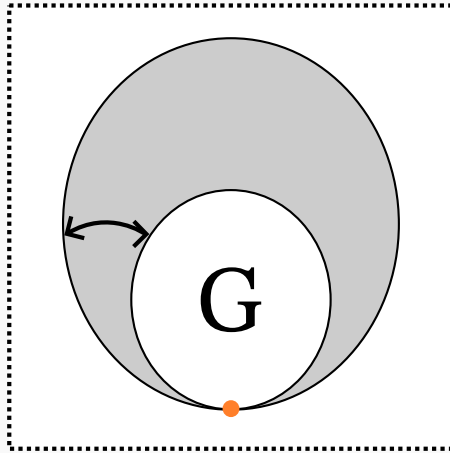
Opening



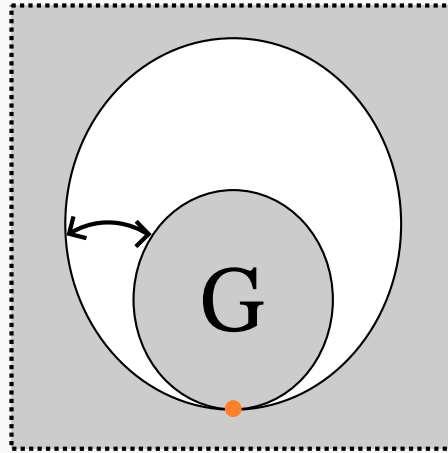
Closing

Interactions

Opening

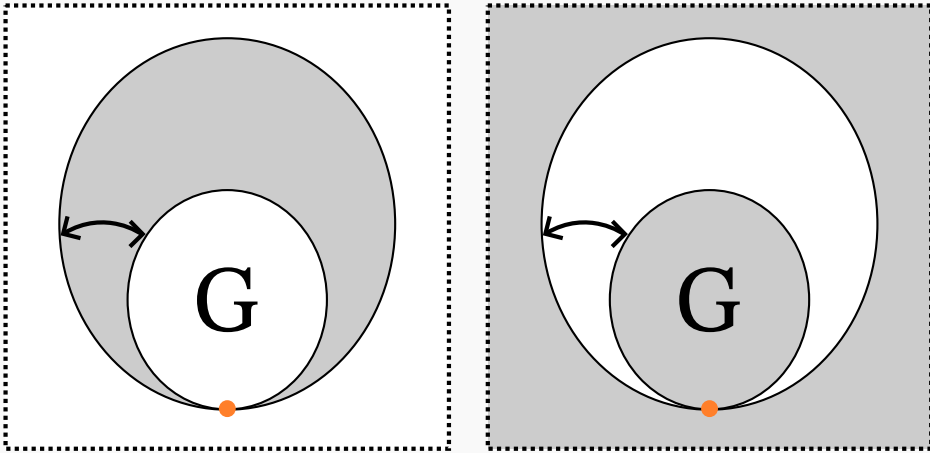


Closing

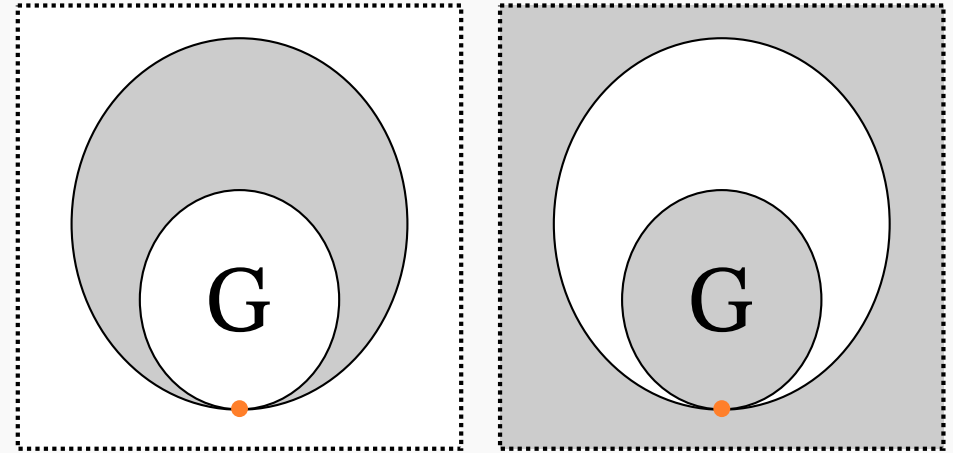


Interactions

Opening

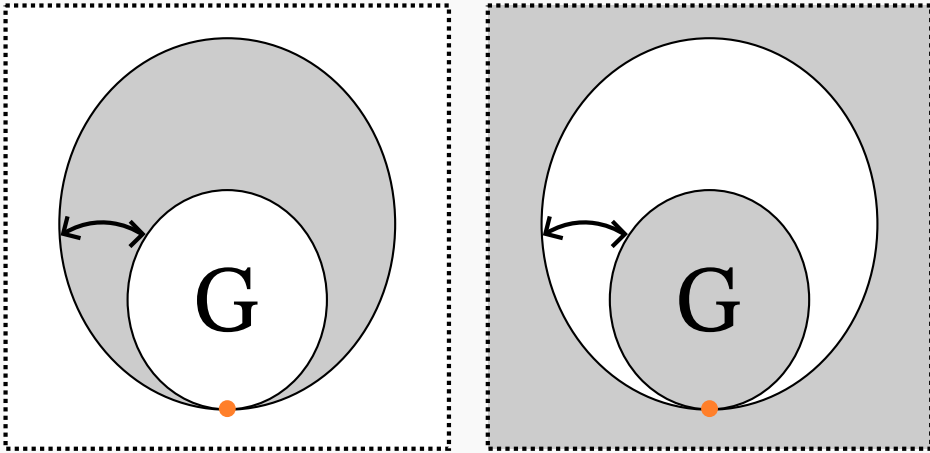


Closing

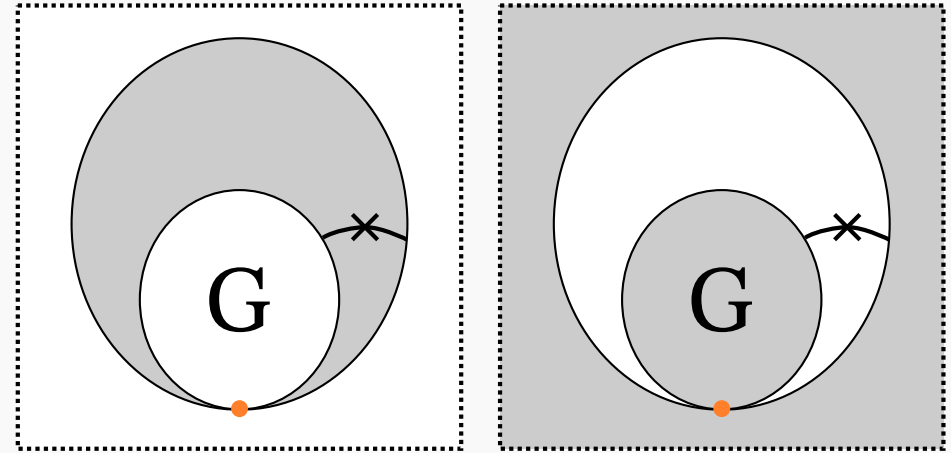


Interactions

Opening



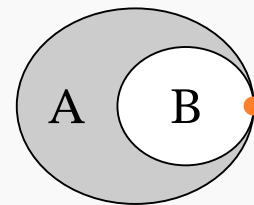
Closing



Simply-typed λ -calculus

Simply-typed λ -calculus

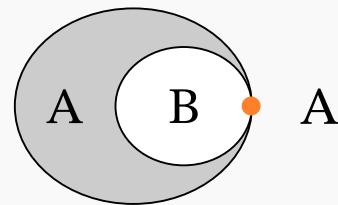
$$\frac{}{x : A, f : A \rightarrow B \vdash f : A \rightarrow B} \text{var}$$



Simply-typed λ -calculus

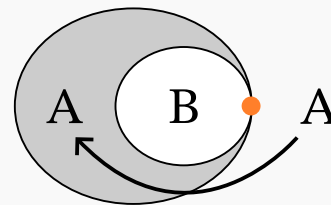
$$\frac{}{x : A, f : A \rightarrow B \vdash f : A \rightarrow B} \text{var}$$

$$\frac{}{x : A, f : A \rightarrow B \vdash x : A} \text{var}$$



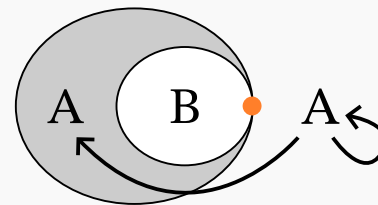
Simply-typed λ -calculus

$$\frac{\frac{}{x:A, f:A \rightarrow B \vdash f:A \rightarrow B} \text{var} \quad \frac{}{x:A, f:A \rightarrow B \vdash x:A} \text{var}}{x:A, f:A \rightarrow B \vdash (f)x:B} \text{app}$$



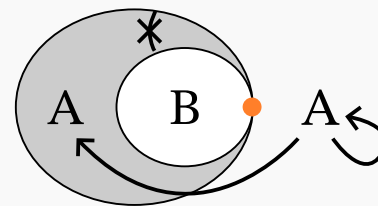
Simply-typed λ -calculus

$$\frac{\frac{}{x:A, f:A \rightarrow B \vdash f:A \rightarrow B} \text{var} \quad \frac{}{x:A, f:A \rightarrow B \vdash x:A} \text{var}}{x:A, f:A \rightarrow B \vdash (f)x:B} \text{app}$$



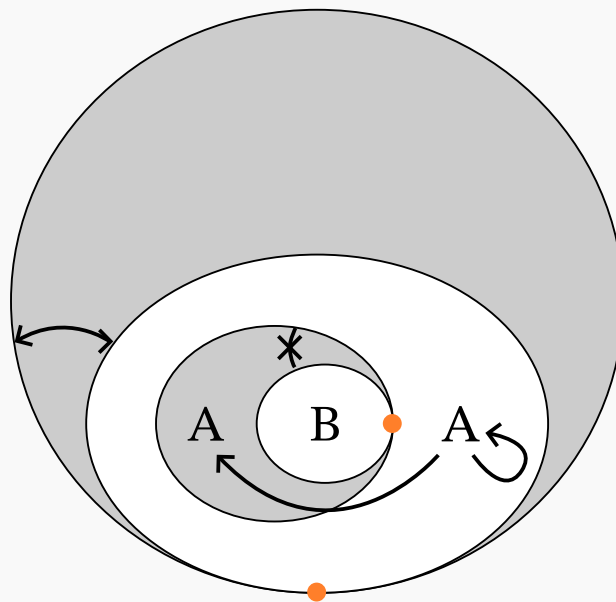
Simply-typed λ -calculus

$$\frac{\frac{}{x:A, f:A \rightarrow B \vdash f:A \rightarrow B} \text{var} \quad \frac{}{x:A, f:A \rightarrow B \vdash x:A} \text{var}}{x:A, f:A \rightarrow B \vdash (f)x:B} \text{app}$$



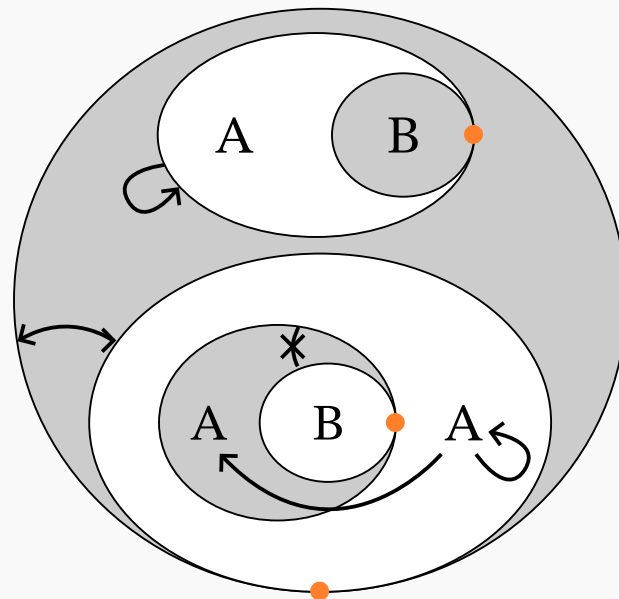
Simply-typed λ -calculus

$$\frac{}{x : A, f : A \rightarrow B \vdash f : A \rightarrow B} \text{var} \quad \frac{}{x : A, f : A \rightarrow B \vdash x : A} \text{var}$$
$$\frac{}{x : A \vdash \lambda f. (f)x : (A \rightarrow B) \rightarrow B} \text{lam}$$
$$\frac{x : A, f : A \rightarrow B \vdash (f)x : B}{x : A \vdash \lambda f. (f)x : (A \rightarrow B) \rightarrow B} \text{app}$$



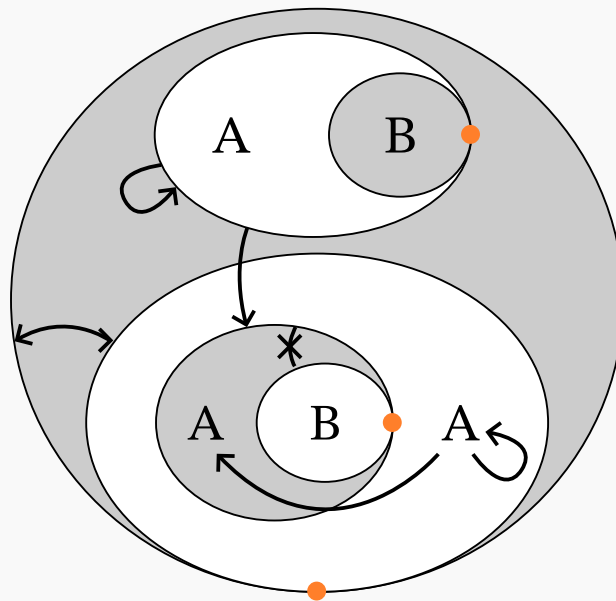
Simply-typed λ -calculus

$$\begin{array}{c}
 \frac{}{x : A, f : A \rightarrow B \vdash f : A \rightarrow B} \text{var} \qquad \frac{}{x : A, f : A \rightarrow B \vdash x : A} \text{var} \\
 \hline
 \frac{x : A, f : A \rightarrow B \vdash (f)x : B}{x : A \vdash \lambda f. (f)x : (A \rightarrow B) \rightarrow B} \text{lam} \qquad \text{app}
 \end{array}$$



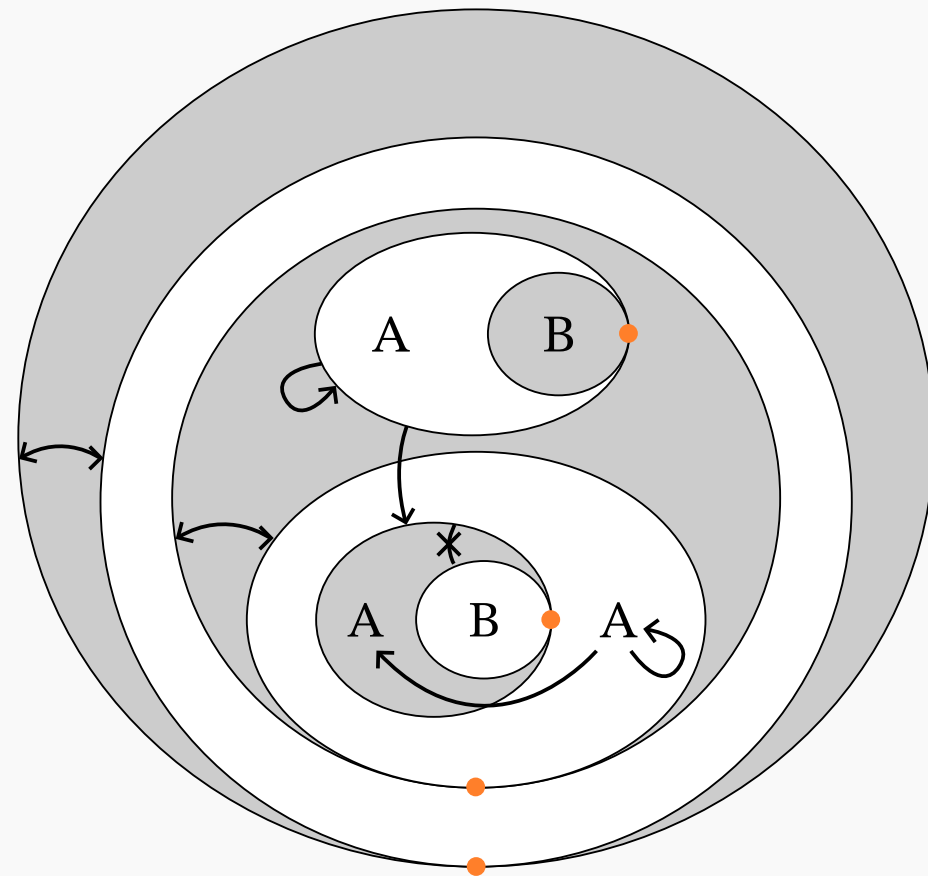
Simply-typed λ -calculus

$$\frac{}{x : A, f : A \rightarrow B \vdash f : A \rightarrow B} \text{var} \quad \frac{}{x : A, f : A \rightarrow B \vdash x : A} \text{var}$$
$$\frac{x : A, f : A \rightarrow B \vdash (f)x : B}{x : A \vdash \lambda f. (f)x : (A \rightarrow B) \rightarrow B} \text{lam}$$
$$\frac{}{x : A, f : A \rightarrow B \vdash (f)x : B} \text{app}$$



Simply-typed λ -calculus

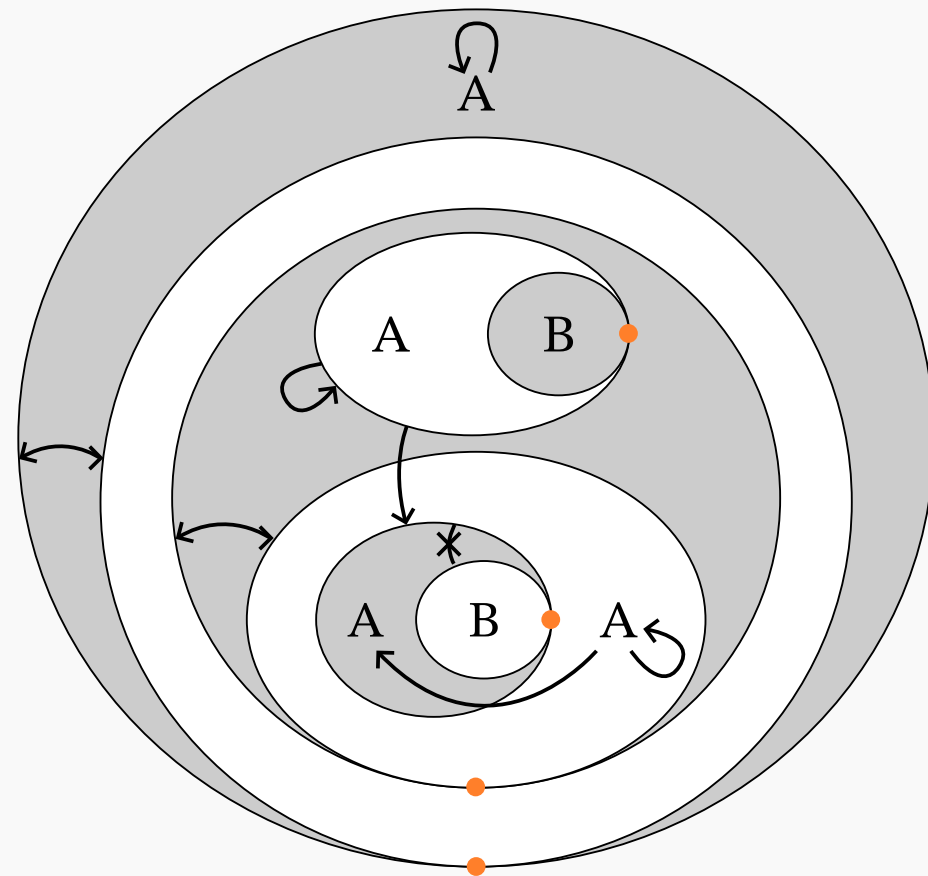
$$\begin{array}{c}
 \frac{}{x : A, f : A \rightarrow B \vdash f : A \rightarrow B} \text{var} \quad \frac{}{x : A, f : A \rightarrow B \vdash x : A} \text{var} \\
 \hline
 \frac{}{x : A, f : A \rightarrow B \vdash (f)x : B} \text{app} \\
 \hline
 \frac{x : A, f : A \rightarrow B \vdash (f)x : B}{x : A \vdash \lambda f. (f)x : (A \rightarrow B) \rightarrow B} \text{lam} \\
 \hline
 \frac{x : A \vdash \lambda f. (f)x : (A \rightarrow B) \rightarrow B}{\vdash \lambda x. \lambda f. (f)x : A \rightarrow (A \rightarrow B) \rightarrow B} \text{lam}
 \end{array}$$



Simply-typed λ -calculus

$$\begin{array}{c}
 \frac{}{x : A, f : A \rightarrow B \vdash f : A \rightarrow B} \text{var} \qquad \frac{}{x : A, f : A \rightarrow B \vdash x : A} \text{var} \\
 \hline
 \frac{x : A, f : A \rightarrow B \vdash (f)x : B}{x : A \vdash \lambda f.(f)x : (A \rightarrow B) \rightarrow B} \text{lam} \\
 \hline
 \frac{}{\vdash \lambda x.\lambda f.(f)x : A \rightarrow (A \rightarrow B) \rightarrow B} \text{lam}
 \end{array}$$

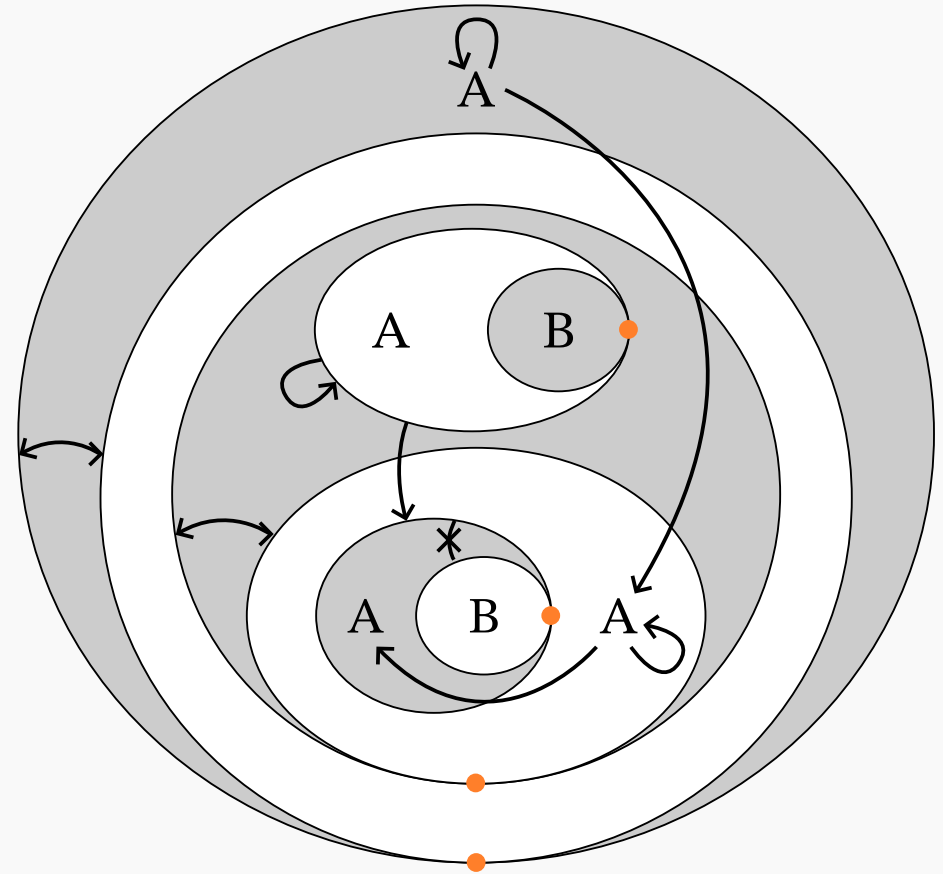
app



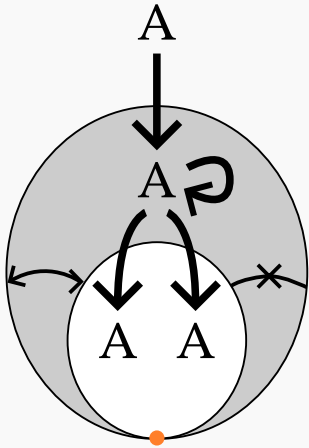
Simply-typed λ -calculus

$$\begin{array}{c}
 \frac{}{x : A, f : A \rightarrow B \vdash f : A \rightarrow B} \text{var} \qquad \frac{}{x : A, f : A \rightarrow B \vdash x : A} \text{var} \\
 \hline
 \frac{x : A, f : A \rightarrow B \vdash (f)x : B}{x : A \vdash \lambda f.(f)x : (A \rightarrow B) \rightarrow B} \text{lam} \\
 \hline
 \frac{}{\vdash \lambda x.\lambda f.(f)x : A \rightarrow (A \rightarrow B) \rightarrow B} \text{lam}
 \end{array}$$

app

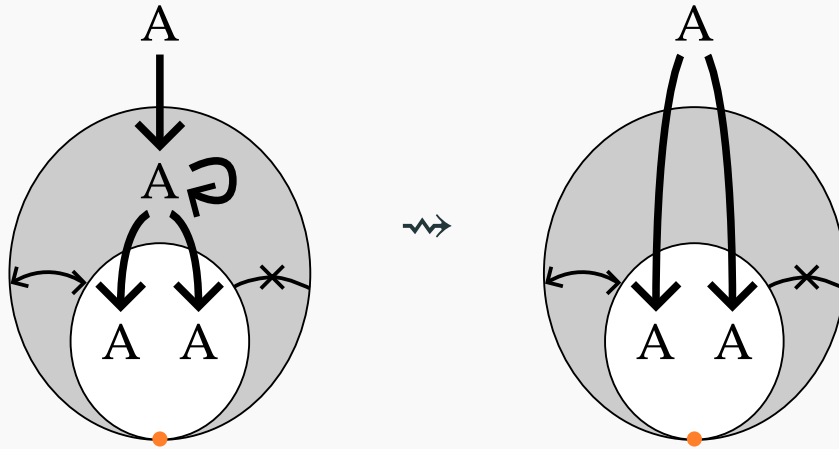


Simulating β -reduction



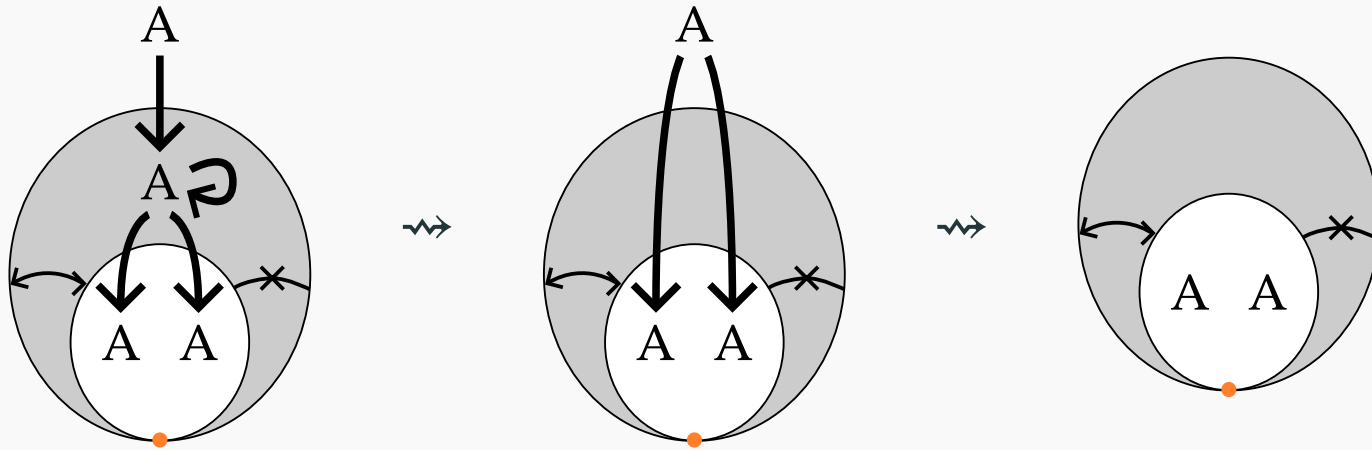
$(\lambda x. \langle x, x \rangle) y$

Simulating β -reduction



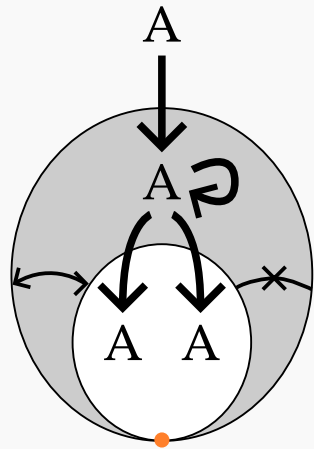
$(\lambda x. \langle x, x \rangle) y$

Simulating β -reduction

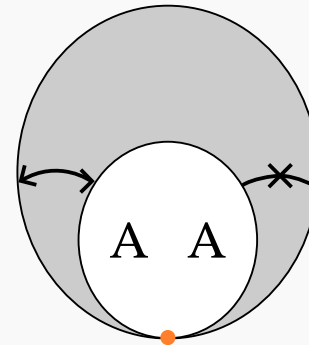
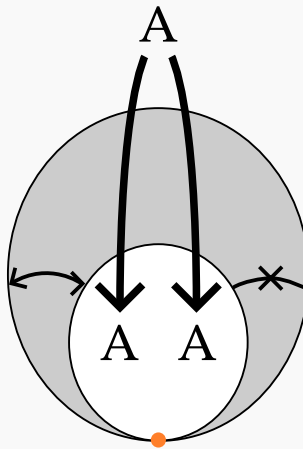


$(\lambda x. \langle x, x \rangle) y$

Simulating β -reduction



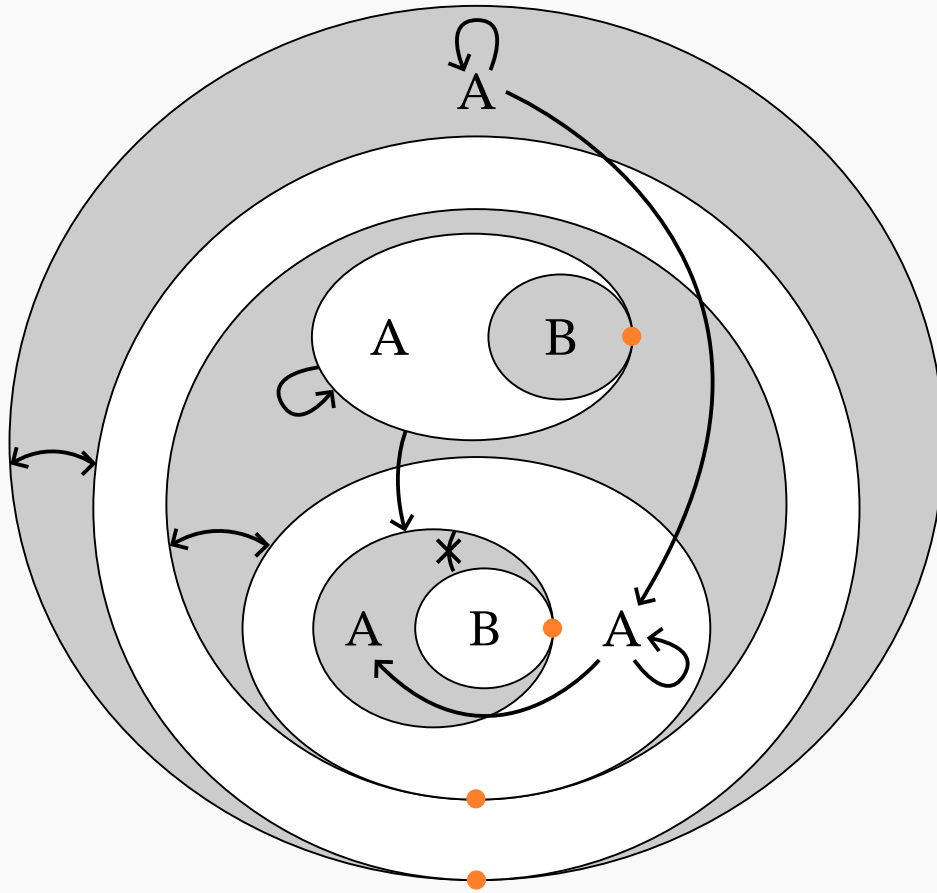
$(\lambda x. \langle x, x \rangle) y$



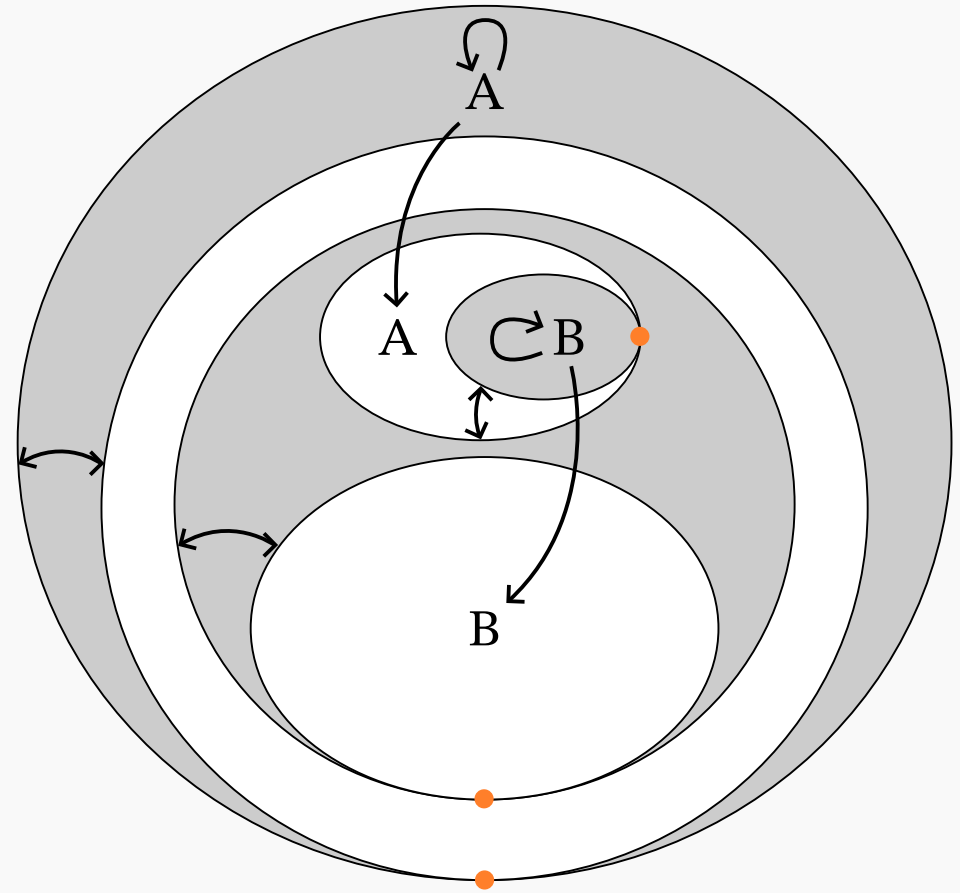
$A \ A$

$\langle y, y \rangle$

Function call vs. inlining



$\vdash A \rightarrow (A \rightarrow B) \rightarrow B$



$\vdash A \rightarrow (A \rightarrow B) \rightarrow B$

[TODO] Computational expressivity

- Propositional logic \simeq **non-recursive, pure** functional programming:
 - Functions (\Rightarrow)
 - Non-recursive algebraic datatypes (\wedge, \vee)
- **Real-world** programming by encoding *more expressive types*:
 - **(Co)inductive types**: (co)recursion
 - **Higher-order types**: polymorphism
 - **Dependent types**: type-level computation
 - **Modal types**: (monadic) side-effects?

Logic is about **abstract, generic** interactions

↳ captures well (the structure of) *general-purpose* programming

[TODO] Notational freedom

Logic is about **abstract, generic** interactions

↳ captures well (the structure of) *general-purpose* programming

BUT (and contrary to popular belief)

most **maths/programming** is about **concrete** representations of the world!

↳ need for *domain-specific* interactive notations

Related works (non-exhaustive)

- **Programming systems:**

- **Boxer** (di Sessa 1994): building programs by manipulating nested boxes
- **Managed copy & paste** (Edwards and Petricek 2022): (de)iteration rules of EGs?
- **Schema evolution** (Edwards et al. 2024):

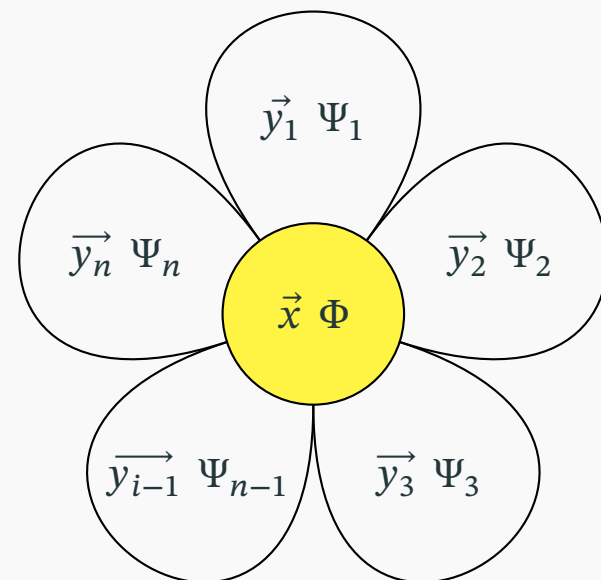
$\text{datatypes} \leftrightarrow \text{logical statements}$
$\text{schema evolution} \overset{?}{\leftrightarrow} \text{illative transformations}$

- **Proof assistants:**

- (Ayers 2021): Box datastructure similar to flowers

- **Categorical logic:**

- (Johnstone 2002): coherent/geometric sequents in topos theory
- (Bonchi et al. 2024): categorical algebra of Beta



$$\forall \vec{x}. \left(\bigwedge \Phi \Rightarrow \bigvee_i \exists \vec{y}_i. \Psi_i \right)$$

Bibliography

- Ayers, Edward W. 2021. “A Tool for Producing Verified, Explainable Proofs..”
- Bertot, Yves, Gilles Kahn, and Laurent Théry. 1994. “Proof by Pointing”. Edited by Masami Hagiya, John C. Mitchell, Gerhard Goos, and Juris Hartmanis. *Theoretical Aspects of Computer Software*. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-57887-0_94.
- Bonchi, Filippo, Alessandro Di Giorgio, Nathan Haydon, and Pawel Sobocinski. 2024. “Diagrammatic Algebra of First Order Logic”. arXiv. January 2024. <https://doi.org/10.48550/arXiv.2401.07055>.
- Chaudhuri, Kaustuv. 2013. “Subformula Linking as an Interaction Method”. Edited by Sandrine Blazy, Christine Paulin-Mohring, David Pichardie, David Hutchinson, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, et al.. *Interactive Theorem*

Proving. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-39634-2_28.

Edwards, Jonathan, and Tomas Petricek. 2022. “Interaction Vs. Abstraction: Managed Copy and Paste”. In *Proceedings of the 1st ACM SIGPLAN International Workshop on Programming Abstractions and Interactive Notations, Tools, And Environments*, 11–19. Auckland New Zealand: ACM. <https://doi.org/10.1145/3563836.3568723>.

Edwards, Jonathan, Tomas Petricek, Tijs Van Der Storm, and Geoffrey Litt. 2024. “Schema Evolution in Interactive Programming Systems”. *The Art, Science, And Engineering of Programming* 9 (1): 2. <https://doi.org/10.22152/programming-journal.org/2025/9/2>.

Johnstone, Peter T. 2002. *Sketches of an Elephant: A Topos Theory Compendium*. Vol. 2. Oxford Logic Guides. Oxford, England: Clarendon Press.

Lewis, C. I. 1920. “A Survey of Symbolic Logic”. *Journal of Philosophy, Psychology and Scientific Methods* 17 (3): 78–79. <https://doi.org/10.2307/2940631>.

Oostra, Arnold. 2010. *Los Gráficos Alfa De Peirce Aplicados a La Lógica Intuicionista*. Cuadernos De Sistemática Peirceana. Centro de Sistemática Peirceana.

Peirce, Charles Sanders. 1906. “Prolegomena to an Apology for Pragmatism”. *The Monist* 16 (4): 492–546. <https://www.jstor.org/stable/27899680>.

Peirce, Charles Sanders. 1976. “Mathematical Miscellanea. 1”. Edited by Carolyn Eisele. *New Elements of Mathematics*. De Gruyter.

Quine, Willard Van Orman. 1955. *Mathematical Logic*. Harvard University Press.

Sessa, A. di. 1994. “Boxer Structures.”