

# Category Theory

Andrew Pitts

University of Cambridge  
Computer Science Tripos  
Part II Unit of Assessment  
Part III and MPhil. ACS Module L108  
Michaelmas Term 2021

# Course web page

Go to

<https://www.cl.cam.ac.uk/teaching/2122/CatTheory/>

<https://www.cl.cam.ac.uk/teaching/2122/L108/>

for

- ▶ these slides and lecture recordings
- ▶ exercise sheets and details of examples classes  
(trying the exercises is essential!)
- ▶ pointers to some additional material

Recommended text for the course is:

[Awodey] Steve Awodey, *Category theory*,  
Oxford University Press (2nd ed.), 2010.

# Assessment

- ▶ **A graded exercise sheet** (25% of the final mark).  
issued in lecture 10 with a one week deadline
- ▶ **A take-home test** (75% of the final mark).  
issued after the end of the course

There will be two one-hour example class sessions to provide help with the exercises.

**See course web page for dates and deadlines.**

# Links to the lectures

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

# Lecture 1

# What is category theory?

What we are probably seeking is a “purer” view of **functions**: a theory of functions in themselves, not a theory of functions derived from sets. What, then, is a pure theory of functions? Answer: **category theory**.

Dana Scott, *Relating theories of the  $\lambda$ -calculus*, p406

**set theory** gives an “element-oriented” account of mathematical structure, whereas

**category theory** takes a ‘function-oriented’ view – understand structures not via their elements, but by how they transform, i.e. via **morphisms**.

(Both theories are part of Logic, broadly construed.)

# GENERAL THEORY OF NATURAL EQUIVALENCES

BY

SAMUEL EILENBERG AND SAUNDERS MACLANE

## CONTENTS

	Page
Introduction .....	231
I. Categories and functors .....	237
1. Definition of categories .....	237
2. Examples of categories .....	239
3. Functors in two arguments .....	241
4. Examples of functors .....	242
5. Slicing of functors .....	245
6. Foundations .....	246
II. Natural equivalence of functors .....	248
7. Transformations of functors .....	248
8. Categories of functors .....	250
9. Composition of functors .....	250
10. Examples of transformations .....	251
11. Groups as categories .....	256
12. Construction of functors by transformations .....	257
13. Combination of the arguments of functors .....	258
III. Functors and groups .....	260
14. Subfunctors .....	260
15. Quotient functors .....	262
16. Examples of subfunctors .....	263
17. The isomorphism theorems .....	265
18. Direct products of functors .....	267
19. Characters .....	270
IV. Partially ordered sets and projective limits .....	272
20. Quasi-ordered sets .....	272
21. Direct systems as functors .....	273
22. Inverse systems as functors .....	276
23. The categories $\mathfrak{Dir}$ and $\mathfrak{Inv}$ .....	277
24. The lifting principle .....	280
25. Functors which commute with limits .....	281
V. Applications to topology .....	283
26. Complexes .....	283
27. Homology and cohomology groups .....	284
28. Duality .....	287
29. Universal coefficient theorems .....	288
30. Čech homology groups .....	290
31. Miscellaneous remarks .....	292
Appendix. Representations of categories .....	292

**Introduction.** The subject matter of this paper is best explained by an example, such as that of the relation between a vector space  $L$  and its "dual"

Presented to the Society, September 8, 1942; received by the editors May 15, 1945.

# Category Theory emerges

1945 Eilenberg<sup>†</sup> and MacLane<sup>†</sup>  
*General Theory of Natural Equivalences*,  
Trans AMS 58, 231–294

(algebraic topology, abstract algebra)

1950s Grothendieck<sup>†</sup> (algebraic geometry)

1960s Lawvere (logic and foundations)

1970s Joyal and Tierney<sup>†</sup> (elementary topos theory)

1980s Dana Scott, Plotkin

(semantics of programming languages)

Lambek<sup>†</sup> (linguistics)



# Category Theory and Computer Science

“Category theory has...become part of the standard “tool-box” in many areas of theoretical informatics, from programming languages to automata, from process calculi to Type Theory.”

Dagstuhl Perspectives Workshop on *Categorical Methods at the Crossroads*  
April 2014

See <http://www.appliedcategorytheory.org/events> for recent examples of category theory being applied (not just in computer science).

# This course

basic concepts of category theory

**adjunction** ← **natural transformation**

**category** → **functor**

applied to { **typed lambda-calculus**  
**functional programming**

# Definition

A **category**  $\mathbf{C}$  is specified by

- ▶ a set  $\text{obj } \mathbf{C}$  whose elements are called **C-objects**
- ▶ for each  $X, Y \in \text{obj } \mathbf{C}$ , a set  $\mathbf{C}(X, Y)$  whose elements are called **C-morphisms from  $X$  to  $Y$**
- ▶ a function assigning to each  $X \in \text{obj } \mathbf{C}$  an element  $\text{id}_X \in \mathbf{C}(X, X)$  called the **identity morphism** for the **C-object**  $X$
- ▶ a function assigning to each  $f \in \mathbf{C}(X, Y)$  and  $g \in \mathbf{C}(Y, Z)$  (where  $X, Y, Z \in \text{obj } \mathbf{C}$ ) an element  $g \circ f \in \mathbf{C}(X, Z)$  called the **composition** of **C-morphisms**  $f$  and  $g$  and satisfying...

# Definition, continued

satisfying...

- ▶ **associativity**: for all  $X, Y, Z, W \in \text{obj } \mathbf{C}$ ,  
 $f \in \mathbf{C}(X, Y)$ ,  $g \in \mathbf{C}(Y, Z)$  and  $h \in \mathbf{C}(Z, W)$

$$h \circ (g \circ f) = (h \circ g) \circ f$$

- ▶ **unity**: for all  $X, Y \in \text{obj } \mathbf{C}$  and  $f \in \mathbf{C}(X, Y)$

$$\text{id}_Y \circ f = f = f \circ \text{id}_X$$


# Example: category of sets, **Set**

- ▶ **obj Set** = some fixed universe of sets

(more on universes later)

- ▶ **Set**( $X, Y$ ) =

$\{f \subseteq X \times Y \mid f \text{ is single-valued and total}\}$



**Cartesian product** of sets  $X$  and  $Y$  is the set of all ordered pairs  $(x, y)$  with  $x \in X$  and  $y \in Y$ .

Equality of ordered pairs:

$$(x, y) = (x', y') \Leftrightarrow x = x' \wedge y = y'$$

# Example: category of sets, **Set**

- ▶ **obj Set** = some fixed universe of sets

(more on universes later)

- ▶ **Set**( $X, Y$ ) =

$\{f \subseteq X \times Y \mid f \text{ is single-valued and total}\}$

$$\forall x \in X, \forall y, y' \in Y, \\ (x, y) \in f \wedge (x, y') \in f \Rightarrow y = y'$$

$$\forall x \in X, \exists y \in Y, \\ (x, y) \in f$$

# Example: category of sets, **Set**

- ▶ **obj Set** = some fixed universe of sets  
(more on universes later)
- ▶ **Set**( $X, Y$ ) =  
 $\{f \subseteq X \times Y \mid f \text{ is single-valued and total}\}$
- ▶  $\text{id}_X = \{(x, x) \mid x \in X\}$
- ▶ composition of  $f \in \text{Set}(X, Y)$  and  $g \in \text{Set}(Y, Z)$  is

$$g \circ f = \{(x, z) \mid \exists y \in Y, (x, y) \in f \wedge (y, z) \in g\}$$

(check that associativity and unity properties hold)

## Example: category of sets, **Set**

**Notation.** Given  $f \in \mathbf{Set}(X, Y)$  and  $x \in X$ , it is usual to write  $f x$  (or  $f(x)$ ) for the unique  $y \in Y$  with  $(x, y) \in f$ .

Thus

$$\begin{aligned}\mathrm{id}_X x &= x \\ (g \circ f) x &= g(f x)\end{aligned}$$



# Domain and codomain

Given a category  $\mathbf{C}$ ,

write  $f : X \rightarrow Y$  or  $X \xrightarrow{f} Y$

to mean that  $f \in \mathbf{C}(X, Y)$ ,

in which case one says

object  $X$  is the **domain** of the morphism  $f$

object  $Y$  is the **codomain** of the morphism  $f$

and writes

$$X = \text{dom } f \quad Y = \text{cod } f$$

(Which category  $\mathbf{C}$  we are referring to is left implicit with this notation.)

# Commutative diagrams

in a category  $\mathbf{C}$ :

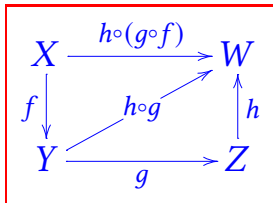
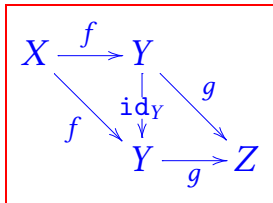
a **diagram** is

a directed graph whose vertices are  $\mathbf{C}$ -objects  
and whose edges are  $\mathbf{C}$ -morphisms

and the diagram is **commutative** (or **commutes**) if  
any two finite paths in the graph between any  
two vertices determine equal morphisms in the  
category under composition

# Commutative diagrams

Examples:



# Alternative notations

I will often just write

$C$  for  $\text{obj } C$

$\text{id}$  for  $\text{id}_X$

Some people write

$\text{Hom}_C(X, Y)$  for  $C(X, Y)$

$1_X$  for  $\text{id}_X$

$gf$  for  $g \circ f$

I use “applicative order” for morphism composition; other people use “diagrammatic order” and write

$f;g$  (or  $f g$ ) for  $g \circ f$

# Alternative definition of category

The definition given here is “dependent-type friendly”.

See [Awodey, Definition 1.1] for an equivalent formulation:

One gives the whole set of morphisms  $\text{mor } \mathbf{C}$   
(in bijection with  $\sum_{X,Y \in \text{obj } \mathbf{C}} \mathbf{C}(X, Y)$  in my definition)  
plus functions

$$\text{dom}, \text{cod} : \text{mor } \mathbf{C} \rightarrow \text{obj } \mathbf{C}$$

$$\text{id} : \text{obj } \mathbf{C} \rightarrow \text{mor } \mathbf{C}$$

and a *partial* function for composition

$$_ \circ _ : \text{mor } \mathbf{C} \times \text{mor } \mathbf{C} \rightarrow \text{mor } \mathbf{C}$$

defined at  $(f, g)$  iff  $\text{cod } f = \text{dom } g$

and satisfying the associativity and unity equations.

## Lecture 2

# Recall

A **category**  $\mathbf{C}$  is specified by

- ▶ a set  $\text{obj } \mathbf{C}$  whose elements are called  **$\mathbf{C}$ -objects**
- ▶ for each  $X, Y \in \text{obj } \mathbf{C}$ , a set  $\mathbf{C}(X, Y)$  whose elements are called  **$\mathbf{C}$ -morphisms from  $X$  to  $Y$**
- ▶ a function assigning to each  $X \in \text{obj } \mathbf{C}$  an element  $\text{id}_X \in \mathbf{C}(X, X)$  called the **identity morphism** for the  $\mathbf{C}$ -object  $X$
- ▶ a function assigning to each  $f \in \mathbf{C}(X, Y)$  and  $g \in \mathbf{C}(Y, Z)$  (where  $X, Y, Z \in \text{obj } \mathbf{C}$ ) an element  $g \circ f \in \mathbf{C}(X, Z)$  called the **composition** of  $\mathbf{C}$ -morphisms  $f$  and  $g$  and satisfying **associativity** and **unity** properties.

# Example:

## category of pre-orders, **Preord**

- ▶ objects are sets  $P$  equipped with a **pre-order**  $_ \sqsubseteq _$   
i.e. a binary relation on  $P$  that is

**reflexive:**  $\forall x \in P, x \sqsubseteq x$

**transitive:**  $\forall x, y, z \in P, x \sqsubseteq y \wedge y \sqsubseteq z \Rightarrow x \sqsubseteq z$

A **partial order** is a pre-order that is also


**anti-symmetric:**  $\forall x, y \in P, x \sqsubseteq y \wedge y \sqsubseteq x \Rightarrow x = y$



# Example:

## category of pre-orders, **Preord**

- ▶ objects are sets  $P$  equipped with a **pre-order**  $\_ \sqsubseteq \_$
- ▶ morphisms: **Preord** $((P_1, \sqsubseteq_1), (P_2, \sqsubseteq_2)) \triangleq$   
 $\{f \in \mathbf{Set}(P_1, P_2) \mid f \text{ is } \mathbf{monotone}\}$


$$\forall x, x' \in P_1, x \sqsubseteq_1 x' \Rightarrow f x \sqsubseteq_2 f x'$$

# Example:

## category of pre-orders, **Preord**

- ▶ objects are sets  $P$  equipped with a **pre-order**  $\sqsubseteq$
- ▶ morphisms:  $\mathbf{Preord}((P_1, \sqsubseteq_1), (P_2, \sqsubseteq_2)) \triangleq \{f \in \mathbf{Set}(P_1, P_2) \mid f \text{ is monotone}\}$
- ▶ identities and composition: as for **Set**

Q: why is this well-defined?

A: because the set of monotone functions contains identity functions and is closed under composition.

# Example:

## category of pre-orders, **Preord**

- ▶ objects are sets  $P$  equipped with a **pre-order**  $\sqsubseteq$
- ▶ morphisms:  $\mathbf{Preord}((P_1, \sqsubseteq_1), (P_2, \sqsubseteq_2)) \triangleq \{f \in \mathbf{Set}(P_1, P_2) \mid f \text{ is monotone}\}$
- ▶ identities and composition: as for **Set**

Pre- and partial orders are relevant to the denotational semantics of programming languages (among other things).

# Example:

## category of monoids, **Mon**

- ▶ objects are **monoids**  $(M, \cdot, e)$  — set  $M$  equipped with a binary operation  $\_ \cdot \_ : M \times M \rightarrow M$  which is **associative**  $\forall x, y, z \in M, x \cdot (y \cdot z) = (x \cdot y) \cdot z$  **has  $e$  as its unit**  $\forall x \in M, e \cdot x = x = x \cdot e$

CS-relevant example of a monoid:  $(\text{List } \Sigma, @, \text{nil})$  where

$\text{List } \Sigma$  = set of finite lists of elements of set  $\Sigma$   
 $@$  = list concatenation  
 $\text{nil} @ \ell = \ell$   
 $(a :: \ell) @ \ell' = a :: (\ell @ \ell')$   
 $\text{nil}$  = empty list

# Example:

## category of monoids, **Mon**

- ▶ objects are **monoids**  $(M, \cdot, e)$
- ▶ morphisms:  $\mathbf{Mon}((M_1, \cdot_1, e_1), (M_2, \cdot_2, e_2)) \triangleq$   
 $\{f \in \mathbf{Set}(M_1, M_2) \mid f e_1 = e_2 \wedge$   
 $\forall x, y \in M_1, f(x \cdot_1 y) = (f x) \cdot_2 (f y)\}$

It's common to denote a monoid  $(M, \cdot, e)$  just by its underlying set  $M$ , leaving  $\cdot$  and  $e$  implicit (hence the same notation gets used for different instances of monoid operations).

# Example:

## category of monoids, **Mon**

- ▶ objects are **monoids**  $(M, \cdot, e)$
- ▶ morphisms:  $\mathbf{Mon}((M_1, \cdot_1, e_1), (M_2, \cdot_2, e_2)) \triangleq$   
 $\{f \in \mathbf{Set}(M_1, M_2) \mid f e_1 = e_2 \wedge$   
 $\quad \forall x, y \in M_1, f(x \cdot_1 y) = (f x) \cdot_2 (f y)\}$
- ▶ identities and composition: as for **Set**

Q: why is this well-defined?

A: because the set of functions that are monoid morphisms contains identity functions and is closed under composition.

# Example:

## category of monoids, **Mon**

- ▶ objects are **monoids**  $(M, \cdot, e)$
- ▶ morphisms:  $\mathbf{Mon}((M_1, \cdot_1, e_1), (M_2, \cdot_2, e_2)) \triangleq$   
 $\{f \in \mathbf{Set}(M_1, M_2) \mid f e_1 = e_2 \wedge$   
 $\forall x, y \in M_1, f(x \cdot_1 y) = (f x) \cdot_2 (f y)\}$
- ▶ identities and composition: as for **Set**

Monoids are relevant to **automata theory** (among other things).

# Example: each pre-order determines a category

Given a pre-ordered set  $(P, \sqsubseteq)$ ,  
we get a category  $\mathbf{C}_P$  by taking

▶ objects  $\text{obj } \mathbf{C}_P = P$

▶ morphisms  $\mathbf{C}_P(x, y) \triangleq \begin{cases} 1 & \text{if } x \sqsubseteq y \\ \emptyset & \text{if } x \not\sqsubseteq y \end{cases}$

(where  $1$  is some fixed one-element set and  $\emptyset$  is the empty set)



# Example: each pre-order determines a category

Given a pre-ordered set  $(P, \sqsubseteq)$ , we get a category  $\mathbf{C}_P$  by taking

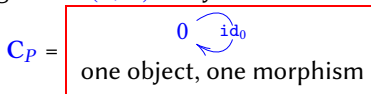
- ▶ objects  $\text{obj } \mathbf{C}_P = P$
- ▶ morphisms  $\mathbf{C}_P(x, y) \triangleq \begin{cases} 1 & \text{if } x \sqsubseteq y \\ \emptyset & \text{if } x \not\sqsubseteq y \end{cases}$
- ▶ identity morphisms and composition are uniquely determined (why?)

# Example: each pre-order determines a category

Given a pre-ordered set  $(P, \sqsubseteq)$ ,  
we get a category  $\mathbf{C}_P$  by taking

- ▶ objects  $\text{obj } \mathbf{C}_P = P$
- ▶ morphisms  $\mathbf{C}_P(x, y) \triangleq \begin{cases} 1 & \text{if } x \sqsubseteq y \\ \emptyset & \text{if } x \not\sqsubseteq y \end{cases}$
- ▶ identity morphisms and composition are uniquely determined (why?)

E.g. when  $(P, \sqsubseteq)$  has just one element  $0$

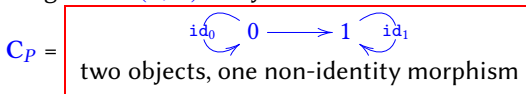


# Example: each pre-order determines a category

Given a pre-ordered set  $(P, \sqsubseteq)$ , we get a category  $\mathbf{C}_P$  by taking

- ▶ objects  $\text{obj } \mathbf{C}_P = P$
- ▶ morphisms  $\mathbf{C}_P(x, y) \triangleq \begin{cases} 1 & \text{if } x \sqsubseteq y \\ \emptyset & \text{if } x \not\sqsubseteq y \end{cases}$
- ▶ identity morphisms and composition are uniquely determined (why?)

E.g. when  $(P, \sqsubseteq)$  has just two elements  $0 \sqsubseteq 1$

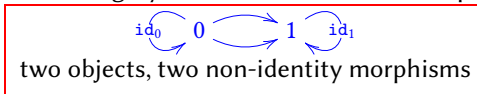


# Example: each pre-order determines a category

Given a pre-ordered set  $(P, \sqsubseteq)$ , we get a category  $\mathbf{C}_P$  by taking

- ▶ objects  $\text{obj } \mathbf{C}_P = P$
- ▶ morphisms  $\mathbf{C}_P(x, y) \triangleq \begin{cases} 1 & \text{if } x \sqsubseteq y \\ \emptyset & \text{if } x \not\sqsubseteq y \end{cases}$
- ▶ identity morphisms and composition are uniquely determined (**why?**)

Example of a finite category that does not arise from a pre-ordered set:



# Example: each monoid determines a category

Given a monoid  $(M, \cdot, e)$ ,  
we get a category  $\mathbf{C}_M$  by taking

- ▶ objects:  $\text{obj } \mathbf{C}_M = 1 = \{0\}$  (one-element set)
- ▶ morphisms:  $\mathbf{C}_M(0, 0) = M$
- ▶ identity morphism:  $\text{id}_0 = e \in M = \mathbf{C}_M(0, 0)$
- ▶ composition of  $f \in \mathbf{C}_M(0, 0)$  and  $g \in \mathbf{C}_M(0, 0)$  is  $g \cdot f \in M = \mathbf{C}_M(0, 0)$

# Definition of isomorphism

Let  $\mathbf{C}$  be a category. A  $\mathbf{C}$ -morphism  $f : X \rightarrow Y$  is an **isomorphism** if there is some  $g : Y \rightarrow X$  for which

$$\begin{array}{ccc} X & \xrightarrow{f} & Y \\ & \searrow \text{id}_X & \downarrow g \\ & & X & \xrightarrow{f} & Y \\ & & & \nearrow \text{id}_Y & \end{array}$$

is a commutative diagram.

# Definition of isomorphism

Let  $\mathbf{C}$  be a category. A  $\mathbf{C}$ -morphism  $f : X \rightarrow Y$  is an **isomorphism** if there is some  $g : Y \rightarrow X$  with  $g \circ f = \text{id}_X$  and  $f \circ g = \text{id}_Y$ .

- ▶ Such a  $g$  is uniquely determined by  $f$  (**why?**) and we write  $f^{-1}$  for it.
- ▶ Given  $X, Y \in \mathbf{C}$ , if such an  $f$  exists, we say the objects  $X$  and  $Y$  are **isomorphic** in  $\mathbf{C}$  and write  $X \cong Y$

(There may be many different  $f$  that witness the fact that  $X$  and  $Y$  are isomorphic.)

**Theorem.** A function  $f \in \mathbf{Set}(X, Y)$  is an isomorphism in the category  $\mathbf{Set}$  iff  $f$  is a bijection, that is

- ▶ **injective:**  $\forall x, x' \in X, f x = f x' \Rightarrow x = x'$
- ▶ **surjective:**  $\forall y \in Y, \exists x \in X, f x = y$

**Proof...**



**Theorem.** A function  $f \in \mathbf{Set}(X, Y)$  is an isomorphism in the category  $\mathbf{Set}$  iff  $f$  is a bijection, that is

- ▶ **injective:**  $\forall x, x' \in X, f x = f x' \Rightarrow x = x'$
- ▶ **surjective:**  $\forall y \in Y, \exists x \in X, f x = y$

Proof...

**Theorem.** A monoid morphism  $f \in \mathbf{Mon}((M_1, \cdot_1, e_1), (M_2, \cdot_2, e_2))$  is an isomorphism in the category  $\mathbf{Mon}$  iff  $f \in \mathbf{Set}(M_1, M_2)$  is a bijection.

Proof...

Define **Poset** to be the category whose objects are **posets** = pre-ordered sets for which the pre-order is anti-symmetric, but is otherwise defined like the category **Preord** of pre-ordered sets.

Define **Poset** to be the category whose objects are **posets** = pre-ordered sets for which the pre-order is anti-symmetric, but is otherwise defined like the category **Preord** of pre-ordered sets.

**Theorem.** A monotone function  $f \in \mathbf{Poset}((P_1, \sqsubseteq_1), (P_2, \sqsubseteq_2))$  is an isomorphism in the category **Poset** iff  $f \in \mathbf{Set}(P_1, P_2)$  is a surjective function satisfying

► **reflective**:  $\forall x, x' \in P_1, f x \sqsubseteq_2 f x' \Rightarrow x \sqsubseteq_1 x'$

**Proof...**

Define **Poset** to be the category whose objects are **posets** = pre-ordered sets for which the pre-order is anti-symmetric, but is otherwise defined like the category **Preord** of pre-ordered sets.

**Theorem.** A monotone function  $f \in \mathbf{Poset}((P_1, \sqsubseteq_1), (P_2, \sqsubseteq_2))$  is an isomorphism in the category **Poset** iff  $f \in \mathbf{Set}(P_1, P_2)$  is a surjective function satisfying

► **reflective:**  $\forall x, x' \in P_1, f x \sqsubseteq_2 f x' \Rightarrow x \sqsubseteq_1 x'$

**Proof...**

(Why does this characterisation not work for **Preord**?)

## Lecture 3

# Category-theoretic properties

Any two isomorphic objects in a category should have the same **category-theoretic properties** – statements that are provable in a formal logic for category theory, whatever that is.

Instead of trying to formalize such a logic, we will just look at examples of category-theoretic properties.

Here is our first one...

# Terminal object

An object  $T$  of a category  $\mathbf{C}$  is **terminal** if for all  $X \in \mathbf{C}$ , there is a unique  $\mathbf{C}$ -morphism from  $X$  to  $T$ , which we write as  $\langle \rangle_X : X \rightarrow T$ .

So we have 
$$\begin{cases} \forall X \in \mathbf{C}, \langle \rangle_X \in \mathbf{C}(X, T) \\ \forall X \in \mathbf{C}, \forall f \in \mathbf{C}(X, T), f = \langle \rangle_X \end{cases}$$

(So in particular,  $\text{id}_T = \langle \rangle_T$ )

Sometimes we just write  $\langle \rangle_X$  as  $\langle \rangle$ .

Some people write  $!_X$  for  $\langle \rangle_X$  – there is no commonly accepted notation; [Awodey] avoids using one.

# Examples of terminal objects

- ▶ In Set: any one-element set.
- ▶ Any one-element set has a unique pre-order and this makes it terminal in Preord (and Poset)
- ▶ Any one-element set has a unique monoid structure and this makes it terminal in Mon.



# Examples of terminal objects

- ▶ In Set: any one-element set.
- ▶ Any one-element set has a unique pre-order and this makes it terminal in Preord (and Poset)
- ▶ Any one-element set has a unique monoid structure and this makes it terminal in Mon.
- ▶ A pre-ordered set  $(P, \sqsubseteq)$ , regarded as a category  $C_P$ , has a terminal object iff it has a **greatest element**  $\top$ , that is:  $\forall x \in P, x \sqsubseteq \top$
- ▶ **When does a monoid  $(M, \cdot, e)$ , regarded as a category  $C_M$ , have a terminal object?**

# Terminal object

**Theorem.** In a category  $\mathbf{C}$ :

- (a) If  $T$  is terminal and  $T \cong T'$ , then  $T'$  is terminal.
- (b) If  $T$  and  $T'$  are both terminal, then  $T \cong T'$  (and there is only one isomorphism between  $T$  and  $T'$ ).

In summary: **terminal objects are unique up to unique isomorphism.**

**Proof...**

# Terminal object

**Theorem.** In a category  $\mathbf{C}$ :

- (a) If  $T$  is terminal and  $T \cong T'$ , then  $T'$  is terminal.
- (b) If  $T$  and  $T'$  are both terminal, then  $T \cong T'$  (and there is only one isomorphism between  $T$  and  $T'$ ).

In summary: **terminal objects are unique up to unique isomorphism.**

**Proof...**

**Notation:** from now on, if a category  $\mathbf{C}$  has a terminal object we will write that object as  $\boxed{1}$

# Opposite of a category

Given a category  $\mathbf{C}$ , its **opposite category**  $\mathbf{C}^{\text{op}}$  is defined by interchanging the operations of **dom** and **cod** in  $\mathbf{C}$ :

- ▶  $\text{obj } \mathbf{C}^{\text{op}} \triangleq \text{obj } \mathbf{C}$
- ▶  $\mathbf{C}^{\text{op}}(X, Y) \triangleq \mathbf{C}(Y, X)$ , for all objects  $X$  and  $Y$
- ▶ identity morphism on  $X \in \text{obj } \mathbf{C}^{\text{op}}$  is  $\text{id}_X \in \mathbf{C}(X, X) = \mathbf{C}^{\text{op}}(X, X)$
- ▶ composition in  $\mathbf{C}^{\text{op}}$  of  $f \in \mathbf{C}^{\text{op}}(X, Y)$  and  $g \in \mathbf{C}^{\text{op}}(Y, Z)$  is given by the composition  $f \circ g \in \mathbf{C}(Z, X) = \mathbf{C}^{\text{op}}(X, Z)$  in  $\mathbf{C}$   
(associativity and unity properties hold for this operation, because they do in  $\mathbf{C}$ )

# The Principle of Duality

Whenever one defines a concept / proves a theorem in terms of commutative diagrams in a category  $\mathbf{C}$ , one obtains another concept / theorem, called its **dual**, by reversing the direction of morphisms throughout, that is, by replacing  $\mathbf{C}$  by its opposite category  $\mathbf{C}^{\text{op}}$ .

For example...

# Initial object

is the dual notion to “terminal object”:

An object  $0$  of a category  $\mathbf{C}$  is **initial** if for all  $X \in \mathbf{C}$ , there is a unique  $\mathbf{C}$ -morphism  $0 \rightarrow X$ , which we write as

$$[\ ]_X : 0 \rightarrow X.$$

So we have 
$$\begin{cases} \forall X \in \mathbf{C}, [\ ]_X \in \mathbf{C}(0, X) \\ \forall X \in \mathbf{C}, \forall f \in \mathbf{C}(0, X), f = [\ ]_X \end{cases}$$

(So in particular,  $\text{id}_0 = [\ ]_0$ )

By duality, we have that initial objects are unique up to isomorphism and that any object isomorphic to an initial object is itself initial.

**(N.B.** “isomorphism” is a self-dual concept.)

# Examples of initial objects

- ▶ The empty set is initial in **Set**.
- ▶ Any one-element set has a uniquely determined monoid structure and is initial in **Mon**. (why?)

So initial and terminal objects co-incide in **Mon**

An object that is both initial and terminal in a category is sometimes called a **zero object**.

- ▶ A pre-ordered set  $(P, \sqsubseteq)$ , regarded as a category  $\mathbf{C}_P$ , has an initial object iff it has a **least element**  $\perp$ , that is:  $\forall x \in P, \perp \sqsubseteq x$

# Example:

## free monoids as initial objects

(relevant to automata and formal languages)

The **free monoid** on a set  $\Sigma$  is  $(\text{List } \Sigma, @, \text{nil})$  where

$\text{List } \Sigma$  = set of finite lists of elements of  $\Sigma$

@ = list concatenation

nil = empty list



# Example:

## free monoids as initial objects

(relevant to automata and formal languages)

The **free monoid** on a set  $\Sigma$  is  $(\text{List } \Sigma, @, \text{nil})$  where

$\text{List } \Sigma$  = set of finite lists of elements of  $\Sigma$

$@$  = list concatenation

$\text{nil}$  = empty list

The function

$\eta_{\Sigma} : \Sigma \rightarrow \text{List } \Sigma$

$a \mapsto [a] = a :: \text{nil}$  (one-element list)

has the following “universal property”...

# Example:

## free monoids as initial objects

(relevant to automata and formal languages)

**Theorem.** For any monoid  $(M, \cdot, e)$  and function  $f : \Sigma \rightarrow M$ , there is a unique monoid morphism  $\bar{f} \in \mathbf{Mon}(\text{List } \Sigma, @, \text{nil}), (M, \cdot, e)$  making

$$\begin{array}{ccc} \Sigma & \xrightarrow{\eta_\Sigma} & \text{List } \Sigma \\ & \searrow f & \downarrow \bar{f} \\ & & M \end{array} \text{ commute in } \mathbf{Set}.$$

**Proof...**

# Example:

## free monoids as initial objects

(relevant to automata and formal languages)

**Theorem.**  $\forall M \in \mathbf{Mon}, \forall f \in \mathbf{Set}(\Sigma, M), \exists! \bar{f} \in \mathbf{Mon}(\mathbf{List} \Sigma, M), \bar{f} \circ \eta_\Sigma = f$

The theorem just says that  $\eta_\Sigma : \Sigma \rightarrow \mathbf{List} \Sigma$  is an initial object in the category  $\Sigma/\mathbf{Mon}$ :

- ▶ objects:  $((M, \cdot, e), f)$  where  $(M, \cdot, e) \in \mathbf{obj} \mathbf{Mon}$  and  $f \in \mathbf{Set}(\Sigma, M)$
- ▶ morphisms in  $\Sigma/\mathbf{Mon}(((M_1, \cdot_1, e_1), f_1), ((M_2, \cdot_2, e_2), f_2))$  are  $f \in \mathbf{Mon}((M_1, \cdot_1, e_1), (M_2, \cdot_2, e_2))$  such that  $f \circ f_1 = f_2$
- ▶ identities and composition as in  $\mathbf{Mon}$

# Example: free monoids as initial objects

(relevant to automata and formal languages)

**Theorem.**  $\forall M \in \mathbf{Mon}, \forall f \in \mathbf{Set}(\Sigma, M), \exists! \bar{f} \in \mathbf{Mon}(\mathbf{List} \Sigma, M), \bar{f} \circ \eta_\Sigma = f$

The theorem just says that  $\eta_\Sigma : \Sigma \rightarrow \mathbf{List} \Sigma$  is an initial object in the category  $\Sigma/\mathbf{Mon}$ :

So this “universal property” determines the monoid  $\mathbf{List} \Sigma$  uniquely up to isomorphism in  $\mathbf{Mon}$ .

We will see later that  $\Sigma \mapsto \mathbf{List} \Sigma$  is part of a functor (= morphism of categories) which is left adjoint to the “forgetful functor”  $\mathbf{Mon} \rightarrow \mathbf{Set}$ .

## Lecture 4

# Binary products

In a category  $\mathbf{C}$ , a **product** for objects  $X, Y \in \mathbf{C}$  is a diagram  $X \xleftarrow{\pi_1} P \xrightarrow{\pi_2} Y$  with the universal property:

For all  $X \xleftarrow{f} Z \xrightarrow{g} Y$  in  $\mathbf{C}$ , there is a unique  $\mathbf{C}$ -morphism  $h : Z \rightarrow P$  such that the following diagram commutes in  $\mathbf{C}$ :

$$\begin{array}{ccc} & Z & \\ f \swarrow & \downarrow h & \searrow g \\ X & P & Y \\ \pi_1 \longleftarrow & & \longrightarrow \pi_2 \end{array}$$

# Binary products

In a category  $\mathbf{C}$ , a **product** for objects  $X, Y \in \mathbf{C}$  is a diagram  $X \xleftarrow{\pi_1} P \xrightarrow{\pi_2} Y$  with the universal property:

For all  $X \xleftarrow{f} Z \xrightarrow{g} Y$  in  $\mathbf{C}$ , there is a unique  $\mathbf{C}$ -morphism  $h : Z \rightarrow P$  such that  $f = \pi_1 \circ h$  and  $g = \pi_2 \circ h$

So  $(P, \pi_1, \pi_2)$  is a terminal object in the category with

- ▶ objects:  $(Z, f, g)$  where  $X \xleftarrow{f} Z \xrightarrow{g} Y$  in  $\mathbf{C}$
- ▶ morphisms  $h : (Z_1, f_1, g_1) \rightarrow (Z_2, f_2, g_2)$  are  $h \in \mathbf{C}(Z_1, Z_2)$  such that  $f_1 = f_2 \circ h$  and  $g_1 = g_2 \circ h$
- ▶ composition and identities as in  $\mathbf{C}$

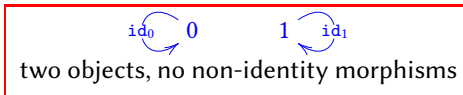
So if it exists, the binary product of two objects in a category is unique up to (unique) isomorphism.

# Binary products

In a category  $\mathbf{C}$ , a **product** for objects  $X, Y \in \mathbf{C}$  is a diagram  $X \xleftarrow{\pi_1} P \xrightarrow{\pi_2} Y$  with the universal property:

For all  $X \xleftarrow{f} Z \xrightarrow{g} Y$  in  $\mathbf{C}$ , there is a unique  $\mathbf{C}$ -morphism  $h : Z \rightarrow P$  such that  $f = \pi_1 \circ h$  and  $g = \pi_2 \circ h$

**N.B.** products of objects in a category do not always exist. For example in the category



the objects 0 and 1 do not have a product, because there is no diagram of the form  $0 \leftarrow ? \rightarrow 1$  in this category.



# Notation for binary products

Assuming  $\mathbf{C}$  has binary products of objects, the product of  $X, Y \in \mathbf{C}$  is written

$$X \xleftarrow{\pi_1} X \times Y \xrightarrow{\pi_2} Y$$

and given  $X \xleftarrow{f} Z \xrightarrow{g} Y$ , the unique  $h : Z \rightarrow X \times Y$  with  $\pi_1 \circ h = f$  and  $\pi_2 \circ h = g$  is written

$$\langle f, g \rangle : Z \rightarrow X \times Y$$

# Examples of products

In **Set**, category-theoretic products are given by the usual cartesian product of sets (set of all ordered pairs)

$$X \times Y = \{(x, y) \mid x \in X \wedge y \in Y\}$$

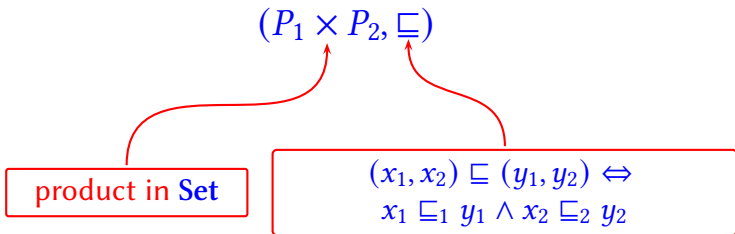
$$\pi_1(x, y) = x$$

$$\pi_2(x, y) = y$$

because...

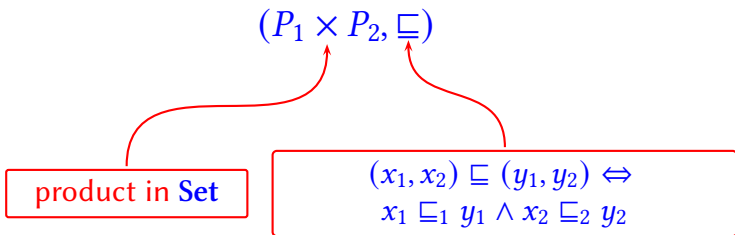
# Examples of products

In **Preord**, can take product of  $(P_1, \sqsubseteq_1)$  and  $(P_2, \sqsubseteq_2)$  to be



# Examples of products

In **Preord**, can take product of  $(P_1, \sqsubseteq_1)$  and  $(P_2, \sqsubseteq_2)$  to be



The projection functions  $P_1 \xleftarrow{\pi_1} P_1 \times P_2 \xrightarrow{\pi_2} P_2$  are monotone for this pre-order on  $P_1 \times P_2$  and have the universal property needed for a product in **Preord** (check).

# Examples of products

In **Mon**, can take product of  $(M_1, \cdot_1, e_1)$  and  $(M_2, \cdot_2, e_2)$  to be

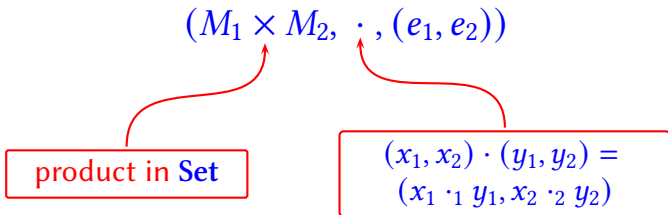
$$(M_1 \times M_2, \cdot, (e_1, e_2))$$

product in **Set**

$$(x_1, x_2) \cdot (y_1, y_2) = \\ (x_1 \cdot_1 y_1, x_2 \cdot_2 y_2)$$

# Examples of products

In **Mon**, can take product of  $(M_1, \cdot_1, e_1)$  and  $(M_2, \cdot_2, e_2)$  to be



The projection functions  $M_1 \xleftarrow{\pi_1} M_1 \times M_2 \xrightarrow{\pi_2} M_2$  are monoid morphisms for this monoid structure on  $M_1 \times M_2$  and have the universal property needed for a product in **Mon** (check).

# Examples of products

Recall that each pre-ordered set  $(P, \sqsubseteq)$  determines a category  $\mathbf{C}_P$ .

Given  $p, q \in P = \text{obj } \mathbf{C}_P$ , the product  $p \times q$  (if it exists) is a **greatest lower bound** (or **glb**, or **meet**) for  $p$  and  $q$  in  $(P, \sqsubseteq)$ :

**lower bound:**

$$p \times q \sqsubseteq p \wedge p \times q \sqsubseteq q$$

**greatest** among all lower bounds:

$$\forall r \in P, r \sqsubseteq p \wedge r \sqsubseteq q \Rightarrow r \sqsubseteq p \times q$$

**Notation:** glbs are often written  $p \wedge q$  or  $p \sqcap q$

# Duality

A binary **coproduct** of two objects in a category  $\mathbf{C}$  is their product in the category  $\mathbf{C}^{\text{op}}$ .



# Duality

A binary **coproduct** of two objects in a category  $\mathbf{C}$  is their product in the category  $\mathbf{C}^{\text{op}}$ .

Thus the coproduct of  $X, Y \in \mathbf{C}$   
if it exists,

is a diagram  $X \xrightarrow{\text{inl}} X + Y \xleftarrow{\text{inr}} Y$   
with the universal property:

$$\forall (X \xrightarrow{f} Z \xleftarrow{g} Y),$$

$$\exists! (X + Y \xrightarrow{h} Z),$$

$$f = h \circ \text{inl} \wedge g = h \circ \text{inr}$$

# Duality

A binary **coproduct** of two objects in a category  $\mathbf{C}$  is their product in the category  $\mathbf{C}^{\text{op}}$ .

E.g. in **Set**, the coproduct of  $X$  and  $Y$

$$X \xrightarrow{\text{inl}} X + Y \xleftarrow{\text{inr}} Y$$

is given by their **disjoint union** (tagged sum)

$$X + Y = \{(0, x) \mid x \in X\} \cup \{(1, y) \mid y \in Y\}$$

$$\text{inl}(x) = (0, x)$$

$$\text{inr}(y) = (1, y)$$

(prove this)

## Lecture 5

# Exponentials

Given  $X, Y \in \mathbf{Set}$ , let  $Y^X \in \mathbf{Set}$  denote the set of all functions from  $X$  to  $Y$ .

$$Y^X = \mathbf{Set}(X, Y) = \{f \subseteq X \times Y \mid f \text{ is single-valued and total}\}$$

Aim to characterise  $Y^X$  category theoretically.

# Exponentials

Given  $X, Y \in \mathbf{Set}$ , let  $Y^X \in \mathbf{Set}$  denote the set of all functions from  $X$  to  $Y$ .

Aim to characterise  $Y^X$  category theoretically.

**Function application** gives a morphism

**app** :  $Y^X \times X \rightarrow Y$  in  $\mathbf{Set}$ .

$$\text{app}(f, x) = f x \quad (f \in Y^X, x \in X)$$

so as a set of ordered pairs, app is  
 $\{(f, x), y) \in (Y^X \times X) \times Y \mid (x, y) \in f\}$

# Exponentials


Given  $X, Y \in \mathbf{Set}$ , let  $Y^X \in \mathbf{Set}$  denote the set of all functions from  $X$  to  $Y$ .

Aim to characterise  $Y^X$  category theoretically.

**Function application** gives a morphism  
 $\text{app} : Y^X \times X \rightarrow Y$  in  $\mathbf{Set}$ .

**Currying** operation transforms morphisms  
 $f : Z \times X \rightarrow Y$  in  $\mathbf{Set}$  to morphisms  $\text{cur } f : Z \rightarrow Y^X$

$$\text{cur } f \ z \ x = f(z, x) \quad (f \in Y^X, z \in Z, x \in X)$$



$$\begin{aligned} \text{cur } f \ z &= \{(x, y) \mid ((z, x), y) \in f\} \\ \text{cur } f &= \{(z, g) \mid g = \{(x, y) \mid ((z, x), y) \in f\}\} \end{aligned}$$

# Haskell Curry

## Haskell Brooks Curry

(/ˈhæskəl/; September 12, 1900 – September 1, 1982)

was an [American mathematician](#) and [logician](#). Curry is best known for his work in [combinatory logic](#); while the initial concept of combinatory logic was based on a single paper by [Moses Schönfinkel](#),<sup>[1]</sup>

much of the development was done by Curry. Curry is also known for [Curry's paradox](#) and the [Curry–Howard correspondence](#).

There are three programming languages named after him, [Haskell](#), [Brook](#) and [Curry](#), as well as the concept of *currying*. a

Haskell Brooks Curry



<b>Born</b>	September 12, 1900 <a href="#">Millis, Massachusetts</a>
<b>Died</b>	September 1, 1982 (aged 81) <a href="#">State College, Pennsylvania</a>
<b>Nationality</b>	American
<b>Alma mater</b>	<a href="#">Harvard University</a>
<b>Known for</b>	<a href="#">Combinatory logic</a> <a href="#">Curry–Howard correspondence</a>

For each function  $f : Z \times X \rightarrow Y$  we get a commutative diagram in **Set**:

$$\begin{array}{ccc}
 Y^X \times X & \xrightarrow{\text{app}} & Y \\
 \text{cur } f \times \text{id}_X \uparrow & & \nearrow f \\
 Z \times X & & \\
 \\ 
 (\text{cur } f \ z, x) & \mapsto & \text{cur } f \ z \ x = f(z, x) \\
 \uparrow & & \nearrow \\
 (z, x) & & 
 \end{array}$$



For each function  $f : Z \times X \rightarrow Y$  we get a commutative diagram in **Set**:

$$\begin{array}{ccc} Y^X \times X & \xrightarrow{\text{app}} & Y \\ \text{cur } f \times \text{id}_X \uparrow & & \nearrow f \\ Z \times X & & \end{array}$$

Furthermore, if any function  $g : Z \rightarrow Y^X$  also satisfies

$$\begin{array}{ccc} Y^X \times X & \xrightarrow{\text{app}} & Y \\ g \times \text{id}_X \uparrow & & \nearrow f \\ Z \times X & & \end{array}$$

then  $g = \text{cur } f$ , because of **function extensionality**...

# Function Extensionality

Two functions  $f, g \in Y^X$  are equal if (and only if)  
 $\forall x \in X, f x = g x.$

This is true of the set-theoretic notion of function, because then

$$\begin{aligned} & \{(x, f x) \mid x \in X\} = \{(x, g x) \mid x \in X\} \\ \text{i.e.} & \quad \{(x, y) \mid (x, y) \in f\} = \{(x, y) \mid (x, y) \in g\} \\ \text{i.e.} & \quad f = g \end{aligned}$$

(in other words it reduces to the extensionality property of sets: two sets are equal iff they have the same elements).

# Exponential objects

Suppose a category **C** has **binary products**, that is, for every pair of **C**-objects  $X$  and  $Y$  there is a product diagram  $X \xleftarrow{\pi_1} X \times Y \xrightarrow{\pi_2} Y$ .

**Notation:** given  $f \in C(X, X')$  and  $f' \in C(Y, Y')$ , then  $f \times f' : X \times Y \rightarrow X' \times Y'$  stands for  $\langle f \circ \pi_1, f' \circ \pi_2 \rangle$ , that is, the unique morphism  $g \in C(X \times Y, X' \times Y')$  satisfying  $\pi_1 \circ g = f \circ \pi_1$  and  $\pi_2 \circ g = f' \circ \pi_2$ .

# Exponential objects

Suppose a category  $\mathbf{C}$  has binary products.

An **exponential** for  $\mathbf{C}$ -objects  $X$  and  $Y$  is specified by

object  $Y^X$  + morphism  $\text{app} : Y^X \times X \rightarrow Y$

satisfying the universal property

for all  $Z \in \mathbf{C}$  and  $f \in \mathbf{C}(Z \times X, Y)$ , there is a unique

$g \in \mathbf{C}(Z, Y^X)$  such that

$$\begin{array}{ccc} Y^X \times X & \xrightarrow{\text{app}} & Y \\ g \times \text{id}_X \uparrow & & \nearrow f \\ Z \times X & & \end{array}$$

commutes in  $\mathbf{C}$ .

**Notation:** we write  $\boxed{\text{cur } f}$  for the unique  $g$  such that  $\text{app} \circ (g \times \text{id}_X) = f$ .

# Exponential objects

The universal property of  $\text{app} : Y^X \times X \rightarrow Y$  says that there is a bijection

$$\mathbf{C}(Z, Y^X) \cong \mathbf{C}(Z \times X, Y)$$

$$g \mapsto \text{app} \circ (g \times \text{id}_X)$$

$$\text{cur } f \leftarrow f$$

$$\text{app} \circ (\text{cur } f \times \text{id}_X) = f$$

$$g = \text{cur}(\text{app} \circ (g \times \text{id}_X))$$

# Exponential objects

The universal property of  $\mathbf{app} : Y^X \times X \rightarrow Y$  says that there is a bijection...

It also says that  $(Y^X, \mathbf{app})$  is a terminal object in the following category:

- ▶ objects:  $(Z, f)$  where  $f \in \mathbf{C}(Z \times X, Y)$
- ▶ morphisms  $g : (Z, f) \rightarrow (Z', f')$  are  $g \in \mathbf{C}(Z, Z')$  such that  $f' \circ (g \times \text{id}_X) = f$
- ▶ composition and identities as in  $\mathbf{C}$ .

So when they exist, exponential objects are unique up to (unique) isomorphism.

# Cartesian closed category

**Definition.**  $\mathcal{C}$  is a **cartesian closed category (ccc)** if it is a category with a terminal object, binary products and exponentials of any pair of objects.

This is a key concept for the semantics of lambda calculus and for the foundations of functional programming languages.

**Notation:** an exponential object  $Y^X$  is often written as  $X \rightarrow Y$

# Cartesian closed category

**Definition.**  $\mathbf{C}$  is a **cartesian closed category (ccc)** if it is a category with a terminal object, binary products and exponentials of any pair of objects.

Examples:

- ▶ **Set** is a ccc — as we have seen.
- ▶ **Preord** is a ccc: we already saw that it has a terminal object and binary products; the exponential of  $(P_1, \sqsubseteq_1)$  and  $(P_2, \sqsubseteq_2)$  is  $(P_1 \rightarrow P_2, \sqsubseteq)$  where

$$P_1 \rightarrow P_2 = \mathbf{Preord}((P_1, \sqsubseteq_1), (P_2, \sqsubseteq_2))$$

$$f \sqsubseteq g \Leftrightarrow \forall x \in P_1, f x \sqsubseteq_2 g x$$

(check that this is a pre-order and does give an exponential in **Preord**)



## Lecture 6

# L108 assessment—heads up

Graded exercise sheet (Ex.Sh.#4) for 25% credit

- ▶ issued Friday 29 October 2021 via Moodle
- ▶ your answers are due (via Moodle) by 12:00 on Friday 5 November 2021

Take-home exam, 75% credit, will be available via Moodle from 12:00 on Friday 26 November 2021, with solutions to be submitted by 12:00 on Friday 3 December 2021.

# CCC

Recall:

**Definition.**  $\mathcal{C}$  is a **cartesian closed category** (ccc) if it is a category with a terminal object, binary products and exponentials of any pair of objects.

# Non-example of a ccc

The category **Mon** of monoids has a terminal object and binary products, but is not a ccc

because of the following bijections between sets, where **1** denotes a one-element set and the corresponding one-element monoid:

$$\begin{aligned} \mathbf{Set}(1, \mathbf{List}\ 1) &\cong \mathbf{Mon}(\mathbf{List}\ 1, \mathbf{List}\ 1) \\ &\cong \mathbf{Mon}(1 \times \mathbf{List}\ 1, \mathbf{List}\ 1) \end{aligned}$$

by universal property of  
the free monoid **List** 1  
on the one-element set **1**

by Ex.Sh. 2, qu. 2  
(**1** is terminal in **Mon**)

# Non-example of a ccc

The category **Mon** of monoids has a terminal object and binary products, but is not a ccc

because of the following bijections between sets, where **1** denotes a one-element set and the corresponding one-element monoid:

$$\begin{aligned}\mathbf{Set}(1, \mathbf{List}\ 1) &\cong \mathbf{Mon}(\mathbf{List}\ 1, \mathbf{List}\ 1) \\ &\cong \mathbf{Mon}(1 \times \mathbf{List}\ 1, \mathbf{List}\ 1)\end{aligned}$$

Since  $\mathbf{Set}(1, \mathbf{List}\ 1)$  is countably infinite, so is  $\mathbf{Mon}(1 \times \mathbf{List}\ 1, \mathbf{List}\ 1)$ .

Since the one-element monoid is initial (see Lect. 3) in **Mon**, for any  $M \in \mathbf{Mon}$  we have that  $\mathbf{Mon}(1, M)$  has just one element and hence

$$\mathbf{Mon}(1 \times \mathbf{List}\ 1, \mathbf{List}\ 1) \neq \mathbf{Mon}(1, M)$$

Therefore no  $M$  can be the exponential of the objects  $\mathbf{List}\ 1$  and  $\mathbf{List}\ 1$  in **Mon**.

# Cartesian closed pre-order

Recall that each pre-ordered set  $(P, \sqsubseteq)$  gives a category  $\mathbf{C}_P$ . It is a ccc iff  $P$  has

- ▶ a **greatest element**  $\top$ :  $\forall p \in P, p \sqsubseteq \top$
- ▶ **binary meets**  $p \wedge q$ :  
 $\forall r \in P, r \sqsubseteq p \wedge q \Leftrightarrow r \sqsubseteq p \wedge r \sqsubseteq q$
- ▶ **Heyting implications**  $p \rightarrow q$ :  
 $\forall r \in P, r \sqsubseteq p \rightarrow q \Leftrightarrow r \wedge p \sqsubseteq q$

# Cartesian closed pre-order

Recall that each pre-ordered set  $(P, \sqsubseteq)$  gives a category  $\mathcal{C}_P$ . It is a ccc iff  $P$  has

- ▶ a **greatest element**  $\top$ :  $\forall p \in P, p \sqsubseteq \top$
- ▶ **binary meets**  $p \wedge q$ :  
 $\forall r \in P, r \sqsubseteq p \wedge q \Leftrightarrow r \sqsubseteq p \wedge r \sqsubseteq q$
- ▶ **Heyting implications**  $p \rightarrow q$ :  
 $\forall r \in P, r \sqsubseteq p \rightarrow q \Leftrightarrow r \wedge p \sqsubseteq q$

E.g. any Boolean algebra (with  $p \rightarrow q = \neg p \vee q$ ).

E.g.  $([0, 1], \leq)$  with  $\top = 1$ ,  $p \wedge q = \min\{p, q\}$  and  $p \rightarrow q = \begin{cases} 1 & \text{if } p \leq q \\ q & \text{if } q < p \end{cases}$

# Intuitionistic Propositional Logic (IPL)

We present it in “natural deduction” style and only consider the fragment with conjunction and implication, with the following syntax:

**Formulas** of IPL:  $\varphi, \psi, \theta, \dots ::=$

$p, q, r, \dots$  propositional identifiers

true truth

$\varphi \ \& \ \psi$  conjunction

$\varphi \Rightarrow \psi$  implication

**Sequents** of IPL:  $\Phi ::= \diamond$  empty

$\Phi, \phi$  non=empty

(so sequents are finite snoc-lists of formulas)



# IPL entailment $\Phi \vdash \varphi$

The intended meaning of  $\Phi \vdash \varphi$  is “the conjunction of the formulas in  $\Phi$  implies the formula  $\varphi$ ”. The relation  $\vdash$  is inductively generated by the following rules:

$\frac{}{\Phi, \varphi \vdash \varphi} \text{ (AX)}$	$\frac{\Phi \vdash \varphi}{\Phi, \psi \vdash \varphi} \text{ (WK)}$	$\frac{\Phi \vdash \varphi \quad \Phi, \varphi \vdash \psi}{\Phi \vdash \psi} \text{ (CUT)}$
$\frac{}{\Phi \vdash \text{true}} \text{ (TRUE)}$	$\frac{\Phi \vdash \varphi \quad \Phi \vdash \psi}{\Phi \vdash \varphi \ \& \ \psi} \text{ (\&I)}$	$\frac{\Phi, \varphi \vdash \psi}{\Phi \vdash \varphi \Rightarrow \psi} \text{ (\Rightarrow I)}$
$\frac{\Phi \vdash \varphi \ \& \ \psi}{\Phi \vdash \varphi} \text{ (\&E}_1\text{)}$	$\frac{\Phi \vdash \varphi \ \& \ \psi}{\Phi \vdash \psi} \text{ (\&E}_2\text{)}$	$\frac{\Phi \vdash \varphi \Rightarrow \psi \quad \Phi \vdash \varphi}{\Phi \vdash \psi} \text{ (\Rightarrow E)}$

For example, if  $\Phi = \diamond$ ,  $\varphi \Rightarrow \psi$ ,  $\psi \Rightarrow \theta$ , then  $\Phi \vdash \varphi \Rightarrow \theta$  is provable in IPL, because:

$$\frac{
 \frac{
 \frac{
 \overline{\Phi \vdash \psi \Rightarrow \theta} \text{ (AX)}
 }{
 \Phi, \varphi \vdash \psi \Rightarrow \theta} \text{ (WK)}
 \quad
 \frac{
 \frac{
 \overline{\diamond, \varphi \Rightarrow \psi \vdash \varphi \Rightarrow \psi} \text{ (AX)}
 }{
 \Phi \vdash \varphi \Rightarrow \psi} \text{ (WK)}
 \quad
 \frac{
 \overline{\Phi, \varphi \vdash \varphi} \text{ (AX)}
 }{
 \Phi, \varphi \vdash \psi} \text{ (}\Rightarrow\text{E)}
 }{
 \Phi, \varphi \vdash \psi} \text{ (}\Rightarrow\text{E)}
 }{
 \Phi, \varphi \vdash \theta} \text{ (}\Rightarrow\text{E)}
 }{
 \Phi \vdash \varphi \Rightarrow \theta} \text{ (}\Rightarrow\text{I)}$$

# Semantics of IPL

in a cartesian closed pre-order  $(P, \sqsubseteq)$

Given a function  $M$  assigning a meaning to each propositional identifier  $p$  as an element  $M(p) \in P$ , we can assign meanings to IPL formula  $\varphi$  and sequents  $\Phi$  as element  $M[\varphi], M[\Phi] \in P$  by recursion on their structure:

$$M[[p]] = M(p)$$

$$M[[\text{true}]] = \top$$

$$M[[\varphi \& \psi]] = M[[\varphi]] \wedge M[[\psi]]$$

$$M[[\varphi \Rightarrow \psi]] = M[[\varphi]] \rightarrow M[[\psi]]$$

$$M[[\diamond]] = \top$$

$$M[[\Phi, \varphi]] = M[[\Phi]] \wedge M[[\varphi]]$$

greatest element

binary meet

Heyting implication

greatest element

binary meet

# Semantics of IPL

in a cartesian closed pre-order  $(P, \sqsubseteq)$

**Soundness Theorem.** If  $\Phi \vdash \varphi$  is provable from the rules of IPL, then  $M[\Phi] \sqsubseteq M[\varphi]$  holds in any cartesian closed pre-order.

**Proof.** *exercise* (show that  $\{(\Phi, \varphi) \mid M[\Phi] \sqsubseteq M[\varphi]\}$  is closed under the rules defining IPL entailment and hence contains  $\{(\Phi, \varphi) \mid \Phi \vdash \varphi\}$ )

# Example

Peirce's Law  $\diamond \vdash ((\varphi \Rightarrow \psi) \Rightarrow \varphi) \Rightarrow \varphi$

is not provable in IPL.

(whereas the formula  $((\varphi \Rightarrow \psi) \Rightarrow \varphi) \Rightarrow \varphi$  is a classical tautology)

# Example

**Peirce's Law**  $\diamond \vdash ((\varphi \Rightarrow \psi) \Rightarrow \varphi) \Rightarrow \varphi$   
is not provable in IPL.

(whereas the formula  $((\varphi \Rightarrow \psi) \Rightarrow \varphi) \Rightarrow \varphi$  is a classical tautology)

For if  $\diamond \vdash ((\varphi \Rightarrow \psi) \Rightarrow \varphi) \Rightarrow \varphi$  were provable in IPL, then by the Soundness Theorem we would have

$$\top = M[\diamond] \sqsubseteq M[((\varphi \Rightarrow \psi) \Rightarrow \varphi) \Rightarrow \varphi].$$

But in the cartesian closed partial order  $([0, 1], \leq)$ , taking  $M(p) = 1/2$  and  $M(q) = 0$ , we get

$$\begin{aligned} M[((p \Rightarrow q) \Rightarrow p) \Rightarrow p] &= ((1/2 \rightarrow 0) \rightarrow 1/2) \rightarrow 1/2 \\ &= (0 \rightarrow 1/2) \rightarrow 1/2 \\ &= 1 \rightarrow 1/2 \\ &= 1/2 \\ &\neq 1 \end{aligned}$$

# Semantics of IPL

in a cartesian closed pre-order  $(P, \sqsubseteq)$

**Completeness Theorem.** Given  $\Phi, \varphi$ , if for all cartesian closed pre-orders  $(P, \sqsubseteq)$  and all interpretations  $M$  of the propositional identifiers as elements of  $P$ , it is the case that  $M[\Phi] \sqsubseteq M[\varphi]$  holds in  $P$ , then  $\Phi \vdash \varphi$  is provable in IPL.

# Semantics of IPL

in a cartesian closed pre-oder  $(P, \sqsubseteq)$

**Completeness Theorem.** Given  $\Phi, \varphi$ , if for all cartesian closed pre-orders  $(P, \sqsubseteq)$  and all interpretations  $M$  of the propositional identifiers as elements of  $P$ , it is the case that  $M[\Phi] \sqsubseteq M[\varphi]$  holds in  $P$ , then  $\Phi \vdash \varphi$  is provable in IPL.

**Proof.** Define

$$P \triangleq \{\text{formulas of IPL}\}$$
$$\varphi \sqsubseteq \psi \triangleq \diamond, \varphi \vdash \psi \text{ is provable in IPL}$$

Then one can show that  $(P, \sqsubseteq)$  is a cartesian closed pre-ordered set.

For this  $(P, \sqsubseteq)$ , taking  $M$  to be  $M(p) = p$ , one can show that  $M[\Phi] \sqsubseteq M[\varphi]$  holds in  $P$  iff  $\Phi \vdash \varphi$  is provable in IPL. □



## Lecture 7

# IPL entailment $\Phi \vdash \varphi$

Recall the rules:

$\frac{}{\Phi, \varphi \vdash \varphi} \text{ (AX)}$	$\frac{\Phi \vdash \varphi}{\Phi, \psi \vdash \varphi} \text{ (WK)}$	$\frac{\Phi \vdash \varphi \quad \Phi, \varphi \vdash \psi}{\Phi \vdash \psi} \text{ (CUT)}$
$\frac{}{\Phi \vdash \text{true}} \text{ (TRUE)}$	$\frac{\Phi \vdash \varphi \quad \Phi \vdash \psi}{\Phi \vdash \varphi \ \& \ \psi} \text{ (\&I)}$	$\frac{\Phi, \varphi \vdash \psi}{\Phi \vdash \varphi \Rightarrow \psi} \text{ (\Rightarrow I)}$
$\frac{\Phi \vdash \varphi \ \& \ \psi}{\Phi \vdash \varphi} \text{ (\&E}_1\text{)}$	$\frac{\Phi \vdash \varphi \ \& \ \psi}{\Phi \vdash \psi} \text{ (\&E}_2\text{)}$	$\frac{\Phi \vdash \varphi \Rightarrow \psi \quad \Phi \vdash \varphi}{\Phi \vdash \psi} \text{ (\Rightarrow E)}$

# Proof theory

Two IPL proofs of  $\diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta \vdash \varphi \Rightarrow \theta$

$$\frac{\frac{\frac{\dots}{\dots} \text{(AX)}}{\Phi, \varphi \vdash \psi \Rightarrow \theta} \text{(WK)} \quad \frac{\frac{\frac{\dots}{\dots} \text{(AX)}}{\Phi, \varphi \vdash \varphi \Rightarrow \psi} \text{(WK)} \quad \frac{\dots}{\Phi, \varphi \vdash \varphi} \text{(AX)}}{\Phi, \varphi \vdash \psi} \text{(}\Rightarrow\text{E)}}{\Phi, \varphi \vdash \theta} \text{(}\Rightarrow\text{I)} \quad \text{where } \Phi \triangleq \diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta$$

$$\frac{\frac{\frac{\dots}{\dots} \text{(AX)}}{\Psi \vdash \varphi \Rightarrow \psi} \text{(WK)} \quad \frac{\dots}{\Psi \vdash \varphi} \text{(AX)}}{\Psi \vdash \psi} \text{(}\Rightarrow\text{E)} \quad \frac{\frac{\frac{\dots}{\dots} \text{(AX)}}{\Psi, \psi \vdash \psi \Rightarrow \theta} \text{(WK)} \quad \frac{\dots}{\Psi, \psi \vdash \psi} \text{(AX)}}{\Psi, \psi \vdash \theta} \text{(}\Rightarrow\text{E)}}{\Psi \vdash \theta} \text{(CUT)} \quad \frac{\Psi \vdash \theta}{\diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta \vdash \varphi \Rightarrow \theta} \text{(}\Rightarrow\text{I)}$$

where  $\Psi \triangleq \diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta, \varphi$

# Proof theory

Two IPL proofs of  $\diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta \vdash \varphi \Rightarrow \theta$

$$\frac{\frac{\frac{\dots}{\dots} (AX)}{\Phi, \varphi \vdash \psi \Rightarrow \theta} (WK) \quad \frac{\frac{\frac{\dots}{\dots} (AX)}{\Phi, \varphi \vdash \varphi \Rightarrow \psi} (WK) \quad \frac{\dots}{\Phi, \varphi \vdash \varphi} (AX)}{\Phi, \varphi \vdash \psi} (\Rightarrow E)}{\Phi, \varphi \vdash \theta} (\Rightarrow E)}{\Phi \vdash \varphi \Rightarrow \theta} (\Rightarrow I) \quad \text{where } \Phi \triangleq \diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta$$

$$\frac{\frac{\frac{\dots}{\dots} (AX)}{\Psi \vdash \varphi \Rightarrow \psi} (WK) \quad \frac{\dots}{\Psi \vdash \varphi} (AX)}{\Psi \vdash \psi} (\Rightarrow E) \quad \frac{\frac{\frac{\dots}{\dots} (AX)}{\Psi, \psi \vdash \psi \Rightarrow \theta} (WK) \quad \frac{\dots}{\Psi, \psi \vdash \psi} (AX)}{\Psi, \psi \vdash \theta} (\Rightarrow E)}{\Psi \vdash \theta} (CUT)}{\diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta \vdash \varphi \Rightarrow \theta} (\Rightarrow I) \quad \text{where } \Psi \triangleq \diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta, \varphi$$

Why is the first proof simpler than the second one?

# Proof theory

$\frac{}{\Phi, \varphi \vdash \varphi} \text{ (AX)}$	$\frac{\Phi \vdash \varphi}{\Phi, \psi \vdash \varphi} \text{ (WK)}$	$\frac{\Phi \vdash \varphi \quad \Phi, \varphi \vdash \psi}{\Phi \vdash \psi} \text{ (CUT)}$
$\frac{}{\Phi \vdash \text{true}} \text{ (TRUE)}$	$\frac{\Phi \vdash \varphi \quad \Phi \vdash \psi}{\Phi \vdash \varphi \& \psi} \text{ (&I)}$	$\frac{\Phi, \varphi \vdash \psi}{\Phi \vdash \varphi \Rightarrow \psi} \text{ (}\Rightarrow\text{I)}$
$\frac{\Phi \vdash \varphi \& \psi}{\Phi \vdash \varphi} \text{ (&E}_1\text{)}$	$\frac{\Phi \vdash \varphi \& \psi}{\Phi \vdash \psi} \text{ (&E}_2\text{)}$	$\frac{\Phi \vdash \varphi \Rightarrow \psi \quad \Phi \vdash \varphi}{\Phi \vdash \psi} \text{ (}\Rightarrow\text{E)}$

**FACT:** if an IPL sequent  $\Phi \vdash \phi$  is provable from the rules, it is provable without using the (CUT) rule.

# Proof theory

$\frac{}{\Phi, \varphi \vdash \varphi} \text{ (AX)}$	$\frac{\Phi \vdash \varphi}{\Phi, \psi \vdash \varphi} \text{ (WK)}$	$\frac{\Phi \vdash \varphi \quad \Phi, \varphi \vdash \psi}{\Phi \vdash \psi} \text{ (CUT)}$
$\frac{}{\Phi \vdash \text{true}} \text{ (TRUE)}$	$\frac{\Phi \vdash \varphi \quad \Phi \vdash \psi}{\Phi \vdash \varphi \& \psi} \text{ (&I)}$	$\frac{\Phi, \varphi \vdash \psi}{\Phi \vdash \varphi \Rightarrow \psi} \text{ (}\Rightarrow\text{I)}$
$\frac{\Phi \vdash \varphi \& \psi}{\Phi \vdash \varphi} \text{ (&E}_1\text{)}$	$\frac{\Phi \vdash \varphi \& \psi}{\Phi \vdash \psi} \text{ (&E}_2\text{)}$	$\frac{\Phi \vdash \varphi \Rightarrow \psi \quad \Phi \vdash \varphi}{\Phi \vdash \psi} \text{ (}\Rightarrow\text{E)}$

**FACT:** if an IPL sequent  $\Phi \vdash \phi$  is provable from the rules, it is provable without using the (CUT) rule.

Simply-Typed Lambda Calculus provides a language for describing proofs in IPL and their properties...

# Simply-Typed Lambda Calculus (STLC)

**Types:**  $A, B, C, \dots ::=$

$G, G', G'' \dots$  “ground” types

$\text{unit}$  unit type

$A \times B$  product type

$A \rightarrow B$  function type

# Simply-Typed Lambda Calculus (STLC)

**Types:**  $A, B, C, \dots ::=$

$G, G', G'' \dots$	“ground” types
$\text{unit}$	unit type
$A \times B$	product type
$A \rightarrow B$	function type

**Terms:**  $s, t, r, \dots ::=$

$c^A$	constants (of given type $A$ )
$x$	variable (countably many)
$()$	unit value
$(s, t)$	pair
$\text{fst } t \quad \text{snd } t$	projections
$\lambda x : A. t$	function abstraction
$s t$	function application



# STLC

Some examples of terms:

- ▶  $\lambda z : (A \rightarrow B) \times (A \rightarrow C). \lambda x : A. ((\text{fst } z) x, (\text{snd } z) x)$   
(has type  $((A \rightarrow B) \times (A \rightarrow C)) \rightarrow (A \rightarrow (B \times C))$ )
- ▶  $\lambda z : A \rightarrow (B \times C). (\lambda x : A. \text{fst}(z x), \lambda y : A. \text{snd}(z y))$   
(has type  $(A \rightarrow (B \times C)) \rightarrow ((A \rightarrow B) \times (A \rightarrow C))$ )
- ▶  $\lambda z : A \rightarrow (B \times C). \lambda x : A. ((\text{fst } z) x, (\text{snd } z) x)$   
(has no type)

# STLC typing relation, $\Gamma \vdash t : A$

$\Gamma$  ranges over **typing environments**

$$\Gamma ::= \diamond \mid \Gamma, x : A$$

(so typing environments are comma-separated snoc-lists of (variable,type)-pairs  
— in fact only the lists whose variables are mutually distinct get used)

The typing relation  $\Gamma \vdash t : A$  is inductively defined by the following rules, which make use of the following notation

$\boxed{\Gamma \text{ ok}}$  means: no variable occurs more than once in  $\Gamma$

$\boxed{\text{dom } \Gamma}$  = finite set of variables occurring in  $\Gamma$

# STLC typing relation, $\Gamma \vdash t : A$

## Typing rules for variables

$$\frac{\Gamma \text{ ok} \quad x \notin \text{dom } \Gamma}{\Gamma, x : A \vdash x : A} \text{ (VAR)}$$

$$\frac{\Gamma \vdash x : A \quad x' \notin \text{dom } \Gamma}{\Gamma, x' : A' \vdash x : A} \text{ (VAR')}$$

## Typing rules for constants and unit value

$$\frac{\Gamma \text{ ok}}{\Gamma \vdash c^A : A} \text{ (CONS)}$$

$$\frac{\Gamma \text{ ok}}{\Gamma \vdash () : \text{unit}} \text{ (UNIT)}$$

# STLC typing relation, $\Gamma \vdash t : A$

## Typing rules for pairs and projections

$$\frac{\Gamma \vdash s : A \quad \Gamma \vdash t : B}{\Gamma \vdash (s, t) : A \times B} \text{ (PAIR)}$$

$$\frac{\Gamma \vdash t : A \times B}{\Gamma \vdash \text{fst } t : A} \text{ (FST)}$$

$$\frac{\Gamma \vdash t : A \times B}{\Gamma \vdash \text{snd } t : B} \text{ (SND)}$$

# STLC typing relation, $\Gamma \vdash t : A$

## Typing rules for function abstraction & application

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x : A. t : A \rightarrow B} \text{ (FUN)}$$

$$\frac{\Gamma \vdash s : A \rightarrow B \quad \Gamma \vdash t : A}{\Gamma \vdash st : B} \text{ (APP)}$$

# STLC typing relation, $\Gamma \vdash t : A$

Example typing derivation:

$$\begin{array}{c}
 \frac{\frac{\frac{\Gamma \vdash g : B \rightarrow C}{\Gamma, x : A \vdash g : B \rightarrow C} \text{(VAR)}}{\Gamma, x : A \vdash g : B \rightarrow C} \text{(VAR')}}{\Gamma, x : A \vdash g(fx) : C} \text{(APP)} \quad \frac{\frac{\frac{\frac{\diamond, f : A \rightarrow B \vdash f : A \rightarrow B}{\Gamma \vdash f : A \rightarrow B} \text{(VAR)}}{\Gamma, x : A \vdash f : A \rightarrow B} \text{(VAR')}}{\Gamma, x : A \vdash fx : B} \text{(APP)} \quad \frac{\Gamma, x : A \vdash x : A}{\Gamma, x : A \vdash x : A} \text{(VAR)}}{\Gamma \vdash \lambda x : A. g(fx) : A \rightarrow C} \text{(FUN)}}{\diamond, f : A \rightarrow B \vdash \lambda g : B \rightarrow C. \lambda x : A. g(fx) : (B \rightarrow C) \rightarrow (A \rightarrow C)} \text{(FUN)}}{\diamond \vdash \lambda f : A \rightarrow B. \lambda g : B \rightarrow C. \lambda x : A. g(fx) : (A \rightarrow B) \rightarrow (B \rightarrow C) \rightarrow (A \rightarrow C)} \text{(FUN)}
 \end{array}$$

where  $\Gamma \triangleq \diamond, f : A \rightarrow B, g : B \rightarrow C$

**N.B.** the STLC typing rules are “syntax-directed”, by the structure of terms  $t$  and then in the case of variables  $x$ , by the structure of typing environments  $\Gamma$ .

# Semantics of STLC types in a ccc

Given a cartesian closed category  $\mathbf{C}$ ,

any function  $M$  mapping ground types  $G$  to objects  $M(G) \in \mathbf{C}$

extends to function  $A \mapsto M[A] \in \mathbf{C}$  and  $\Gamma \mapsto M[\Gamma] \in \mathbf{C}$  from STLC types and typing environments to  $\mathbf{C}$ -objects, by recursion on the structure of  $A$ :

$$M[G] = M(G)$$

$$M[\text{unit}] = 1$$

$$M[A \times B] = M[A] \times M[B]$$

$$M[A \rightarrow B] = M[A] \rightarrow M[B]$$

$$M[\diamond] = 1$$

$$M[\Gamma, x : A] = M[\Gamma] \times M[A]$$

terminal object in  $\mathbf{C}$

product in  $\mathbf{C}$

exponential in  $\mathbf{C}$

terminal object in  $\mathbf{C}$

product in  $\mathbf{C}$

## Lecture 8



# Semantics of STLC terms in a ccc

Given a cartesian closed category  $\mathbf{C}$ ,

given any function  $M$  mapping

- ▶ ground types  $G$  to  $\mathbf{C}$ -objects  $M(G)$   
(which extends to a function mapping all types to objects,  $A \mapsto M[[A]]$ , as we have seen)

# Semantics of STLC terms in a ccc

Given a cartesian closed category  $\mathbf{C}$ ,

given any function  $M$  mapping

- ▶ ground types  $G$  to  $\mathbf{C}$ -objects  $M(G)$
- ▶ constants  $c^A$  to  $\mathbf{C}$ -morphisms  $M(c^A) : 1 \rightarrow M[[A]]$   
(In a category with a terminal object  $1$ , given an object  $X \in \mathbf{C}$ , morphisms  $1 \rightarrow X$  are sometimes called **global elements** of  $X$ .)

# Semantics of STLC terms in a ccc

Given a cartesian closed category  $\mathbf{C}$ ,

given any function  $M$  mapping

- ▶ ground types  $G$  to  $\mathbf{C}$ -objects  $M(G)$
- ▶ constants  $c^A$  to  $\mathbf{C}$ -morphisms  $M(c^A) : 1 \rightarrow M[A]$

we get a function mapping provable instances of the typing relation  $\Gamma \vdash t : A$  to  $\mathbf{C}$ -morphisms

$$M[\Gamma \vdash t : A] : M[\Gamma] \rightarrow M[A]$$

defined by recursing over the proof of  $\Gamma \vdash t : A$  from the typing rules (which follows the structure of  $t$ ):

# Semantics of STLC terms in a ccc

## Variables:

$$M[\Gamma, x : A \vdash x : A] = M[\Gamma] \times M[A] \xrightarrow{\pi_2} M[A]$$

$$M[\Gamma, x' : A' \vdash x : A] =$$

$$M[\Gamma] \times M[A'] \xrightarrow{\pi_1} M[\Gamma] \xrightarrow{M[\Gamma \vdash x : A]} M[A]$$

## Constants:

$$M[\Gamma \vdash c^A : A] = M[\Gamma] \xrightarrow{\langle \rangle} 1 \xrightarrow{M(c^A)} M[A]$$

## Unit value:

$$M[\Gamma \vdash () : \text{unit}] = M[\Gamma] \xrightarrow{\langle \rangle} 1$$

# Semantics of STLC terms in a ccc

## Pairing:

$$M[\Gamma \vdash (s, t) : A \times B] = \\ M[\Gamma] \xrightarrow{\langle M[\Gamma \vdash s : A], M[\Gamma \vdash t : B] \rangle} M[A] \times M[B]$$

## Projections:

$$M[\Gamma \vdash \text{fst } t : A] = \\ M[\Gamma] \xrightarrow{M[\Gamma \vdash t : A \times B]} M[A] \times M[B] \xrightarrow{\pi_1} M[A]$$

# Semantics of STLC terms in a ccc

## Pairing:

$$M[\Gamma \vdash (s, t) : A \times B] = \\ M[\Gamma] \xrightarrow{\langle M[\Gamma \vdash s : A], M[\Gamma \vdash t : B] \rangle} M[A] \times M[B]$$

## Projections:

$$M[\Gamma \vdash \text{fst } t : A] = \\ M[\Gamma] \xrightarrow{M[\Gamma \vdash t : A \times B]} M[A] \times M[B] \xrightarrow{\pi_1} M[A]$$

Given that  $\Gamma \vdash \text{fst } t : A$  holds,  
there is a unique type  $B$   
such that  $\Gamma \vdash t : A \times B$  already  
holds.

**Lemma.** If  $\Gamma \vdash t : A$  and  $\Gamma \vdash t : B$  are provable, then  $A = B$ .

# Semantics of STLC terms in a ccc

## Pairing:

$$M[\Gamma \vdash (s, t) : A \times B] = \\ M[\Gamma] \xrightarrow{\langle M[\Gamma \vdash s : A], M[\Gamma \vdash t : B] \rangle} M[A] \times M[B]$$

## Projections:

$$M[\Gamma \vdash \text{snd } t : B] = \\ M[\Gamma] \xrightarrow{M[\Gamma \vdash t : A \times B]} M[A] \times M[B] \xrightarrow{\pi_2} M[B]$$

(As for the case of `fst`, if  $\Gamma \vdash \text{snd } t : B$ , then  $\Gamma \vdash t : A \times B$  already holds for a unique type  $A$ .)

# Semantics of STLC terms in a ccc

## Function abstraction:

$$M[\Gamma \vdash \lambda x : A. t : A \rightarrow B] = \\ \text{cur } f : M[\Gamma] \rightarrow (M[A] \rightarrow M[B])$$

where

$$f = M[\Gamma, x : A \vdash t : B] : M[\Gamma] \times M[A] \rightarrow M[B]$$



# Semantics of STLC terms in a ccc

## Function application:

$$M[\Gamma \vdash st : B] = \\ M[\Gamma] \xrightarrow{\langle f, g \rangle} (M[A] \rightarrow M[B]) \times M[A] \xrightarrow{\text{app}} M[B]$$

where

$A$  = unique type such that  $\Gamma \vdash s : A \rightarrow B$  and  $\Gamma \vdash t : A$   
already holds (exists because  $\Gamma \vdash st : B$  holds)

$$f = M[\Gamma \vdash s : A \rightarrow B] : M[\Gamma] \rightarrow (M[A] \rightarrow M[B])$$

$$g = M[\Gamma \vdash t : A] : M[\Gamma] \rightarrow M[A]$$

# Example

Consider  $t \triangleq \lambda x : A. g(f x)$  so that  $\Gamma \vdash t : A \rightarrow C$  when  $\Gamma \triangleq \diamond, f : A \rightarrow B, g : B \rightarrow C$ .

Suppose  $M[[A]] = X$ ,  $M[[B]] = Y$  and  $M[[C]] = Z$  in  $\mathbf{C}$ . Then

$$M[[\Gamma]] = (1 \times Y^X) \times Z^Y$$

$$M[[\Gamma, x : A]] = ((1 \times Y^X) \times Z^Y) \times X$$

$$M[[\Gamma, x : A \vdash x : A]] = \pi_2$$

$$M[[\Gamma, x : A \vdash g : B \rightarrow C]] = \pi_2 \circ \pi_1$$

$$M[[\Gamma, x : A \vdash f : A \rightarrow B]] = \pi_2 \circ \pi_1 \circ \pi_1$$

$$M[[\Gamma, x : A \vdash f x : B]] = \mathbf{app} \circ \langle \pi_2 \circ \pi_1 \circ \pi_1, \pi_2 \rangle$$

$$M[[\Gamma, x : A \vdash g(f x) : C]] = \mathbf{app} \circ \langle \pi_2 \circ \pi_1, \mathbf{app} \circ \langle \pi_2 \circ \pi_1 \circ \pi_1, \pi_2 \rangle \rangle$$

$$M[[\Gamma \vdash t : A \rightarrow C]] = \mathbf{cur}(\mathbf{app} \circ \langle \pi_2 \circ \pi_1, \mathbf{app} \circ \langle \pi_2 \circ \pi_1 \circ \pi_1, \pi_2 \rangle \rangle)$$

# STLC equations

take the form  $\Gamma \vdash s = t : A$  where  $\Gamma \vdash s : A$  and  $\Gamma \vdash t : A$  are provable.

Such an equation is **satisfied** by the semantics in a ccc if  $M[\Gamma \vdash s : A]$  and  $M[\Gamma \vdash t : A]$  are equal **C**-morphisms  $M[\Gamma] \rightarrow M[A]$ .

**Qu:** which equations are always satisfied in any ccc?

# STLC equations

take the form  $\Gamma \vdash s = t : A$  where  $\Gamma \vdash s : A$  and  $\Gamma \vdash t : A$  are provable.

Such an equation is **satisfied** by the semantics in a ccc if  $M[\Gamma \vdash s : A]$  and  $M[\Gamma \vdash t : A]$  are equal **C**-morphisms  $M[\Gamma] \rightarrow M[A]$ .

**Qu:** which equations are always satisfied in any ccc?

**Ans:**  **$(\alpha)\beta\eta$ -equivalence** — to define this, first have to define **alpha-equivalence**, **substitution** and its semantics.

# Alpha equivalence of STLC terms

The names of  $\lambda$ -bound variables should not affect meaning.

E.g.  $\lambda f : A \rightarrow B. \lambda x : A. f x$  should have the same meaning as  $\lambda x : A \rightarrow B. \lambda y : A. x y$

# Alpha equivalence of STLC terms

The names of  $\lambda$ -bound variables should not affect meaning.

E.g.  $\lambda f : A \rightarrow B. \lambda x : A. f x$  should have the same meaning as  $\lambda x : A \rightarrow B. \lambda y : A. x y$

This issue is best dealt with at the level of syntax rather than semantics: from now on we re-define “STLC term” to mean not an abstract syntax tree (generated as described before), but rather an equivalence class of such trees with respect to **alpha-equivalence**  $s =_{\alpha} t$ , defined as follows...

(Alternatively, one can use a “nameless” (de Bruijn) representation of terms.)

# Alpha equivalence of STLC terms

$\frac{}{c^A =_\alpha c^A}$	$\frac{}{x =_\alpha x}$	$\frac{}{() =_\alpha ()}$	$\frac{s =_\alpha s' \quad t =_\alpha t'}{(s, t) =_\alpha (s', t')}$	$\frac{t =_\alpha t'}{\text{fst } t =_\alpha \text{fst } t'}$
	$\frac{t =_\alpha t'}{\text{snd } t =_\alpha \text{snd } t'}$		$\frac{s =_\alpha s' \quad t =_\alpha t'}{s t =_\alpha s' t'}$	
$\frac{(y x) \cdot t =_\alpha (y x') \cdot t' \quad y \text{ does not occur in } \{x, x', t, t'\}}{\lambda x : A. t =_\alpha \lambda x' : A. t'}$				

# Alpha equivalence of STLC terms

$\frac{}{c^A =_\alpha c^A}$	$\frac{}{x =_\alpha x}$	$\frac{}{() =_\alpha ()}$	$\frac{s =_\alpha s' \quad t =_\alpha t'}{(s, t) =_\alpha (s', t')}$	$\frac{t =_\alpha t'}{\text{fst } t =_\alpha \text{fst } t'}$
-----------------------------	-------------------------	---------------------------	--	---

$\frac{t =_\alpha t'}{\text{snd } t =_\alpha \text{snd } t'}$	$\frac{s =_\alpha s' \quad t =_\alpha t'}{s t =_\alpha s' t'}$
---	--

$\frac{(y x) \cdot t =_\alpha (y x') \cdot t' \quad y \text{ does not occur in } \{x, x', t, t'\}}{\lambda x : A. t =_\alpha \lambda x' : A. t'}$
---

result of replacing all occurrences of  $x$  with  $y$  in  $t$



# Alpha equivalence of STLC terms

$\frac{}{c^A =_\alpha c^A}$	$\frac{}{x =_\alpha x}$	$\frac{}{() =_\alpha ()}$	$\frac{s =_\alpha s' \quad t =_\alpha t'}{(s, t) =_\alpha (s', t')}$	$\frac{t =_\alpha t'}{\text{fst } t =_\alpha \text{fst } t'}$
	$\frac{t =_\alpha t'}{\text{snd } t =_\alpha \text{snd } t'}$		$\frac{s =_\alpha s' \quad t =_\alpha t'}{s t =_\alpha s' t'}$	
$\frac{(y x) \cdot t =_\alpha (y x') \cdot t' \quad y \text{ does not occur in } \{x, x', t, t'\}}{\lambda x : A. t =_\alpha \lambda x' : A. t'}$				

E.g.

$$\lambda x : A. x x =_\alpha \lambda y : A. y y \neq_\alpha \lambda x : A. x y$$

$$(\lambda y : A. y) x =_\alpha (\lambda x : A. x) x \neq_\alpha (\lambda x : A. x) y$$

# Substitution

$t[s/x]$  = result of replacing all free occurrences of variable  $x$  in term  $t$  (i.e. those not occurring within the scope of a  $\lambda x : A.$  binder) by the term  $s$ , alpha-converting  $\lambda$ -bound variables in  $t$  to avoid them “capturing” any free variables of  $t$ .

E.g.  $(\lambda y : A. (y, x))[y/x]$  is  $\lambda z : A. (z, y)$  and is not  $\lambda y : A. (y, y)$

# Substitution

$t[s/x]$  = result of replacing all free occurrences of variable  $x$  in term  $t$  (i.e. those not occurring within the scope of a  $\lambda x : A.$  binder) by the term  $s$ , alpha-converting  $\lambda$ -bound variables in  $t$  to avoid them “capturing” any free variables of  $t$ .

E.g.  $(\lambda y : A. (y, x))[y/x]$  is  $\lambda z : A. (z, y)$  and is not  $\lambda y : A. (y, y)$

The relation  $t[s/x] = t'$  can be inductively defined by the following rules...

# Substitution

$\frac{}{c^A[s/x] = c^A}$	$\frac{}{x[s/x] = s}$	$\frac{y \neq x}{y[s/x] = y}$	$\frac{}{() [s/x] = ()}$
$\frac{t_1[s/x] = t'_1 \quad t_2[s/x] = t'_2}{(t_1, t_2)[s/x] = (t'_1, t'_2)}$		$\frac{t[s/x] = t'}{(\text{fst } t)[s/x] = \text{fst } t'}$	
$\frac{t[s/x] = t'}{(\text{snd } t)[s/x] = \text{snd } t'}$	$\frac{t_1[s/x] = t'_1 \quad t_2[s/x] = t'_2}{(t_1 t_2)[s/x] = t'_1 t'_2}$		
$\frac{t[s/x] = t' \quad y \neq x \text{ and } y \text{ does not occur in } s}{(\lambda y : A. t)[s/x] = \lambda y : A. t'}$			

# Semantics of substitution in a ccc

**Substitution Lemma** If  $\Gamma \vdash s : A$  and  $\Gamma, x : A \vdash t : B$  are provable, then so is  $\Gamma \vdash t[s/x] : B$ .

**Substitution Theorem** If  $\Gamma \vdash s : A$  and  $\Gamma, x : A \vdash t : B$  are provable, then in any ccc the following diagram commutes:

$$\begin{array}{ccc} M[\Gamma] & \xrightarrow{\langle \text{id}, M[\Gamma \vdash s : A] \rangle} & M[\Gamma] \times M[A] \\ & \searrow^{M[\Gamma \vdash t[s/x] : B]} & \downarrow M[\Gamma, x : A \vdash t : B] \\ & & M[B] \end{array}$$

## Lecture 9

# STLC equations

take the form  $\Gamma \vdash s = t : A$  where  $\Gamma \vdash s : A$  and  $\Gamma \vdash t : A$  are provable.

Such an equation is satisfied by the semantics in a ccc if  $M[\Gamma \vdash s : A]$  and  $M[\Gamma \vdash t : A]$  are equal  $\mathbf{C}$ -morphisms  $M[\Gamma] \rightarrow M[A]$ .

**Qu:** which equations are always satisfied in any ccc?

**Ans:**  $\beta\eta$ -equivalence...

# STLC $\beta\eta$ -Equality

The relation  $\Gamma \vdash s =_{\beta\eta} t : A$  (where  $\Gamma$  ranges over typing environments,  $s$  and  $t$  over terms and  $A$  over types) is inductively defined by the following rules:



# STLC $\beta\eta$ -Equality

The relation  $\Gamma \vdash s =_{\beta\eta} t : A$  (where  $\Gamma$  ranges over typing environments,  $s$  and  $t$  over terms and  $A$  over types) is inductively defined by the following rules:

►  $\beta$ -conversions

$\frac{\Gamma, x : A \vdash t : B \quad \Gamma \vdash s : A}{\Gamma \vdash (\lambda x : A. t)s =_{\beta\eta} t[s/x] : B}$	$\frac{\Gamma \vdash s : A \quad \Gamma \vdash t : B}{\Gamma \vdash \text{fst}(s, t) =_{\beta\eta} s : A}$
$\frac{\Gamma \vdash s : A \quad \Gamma \vdash t : B}{\Gamma \vdash \text{snd}(s, t) =_{\beta\eta} t : B}$	

# STLC $\beta\eta$ -Equality

The relation  $\Gamma \vdash s =_{\beta\eta} t : A$  (where  $\Gamma$  ranges over typing environments,  $s$  and  $t$  over terms and  $A$  over types) is inductively defined by the following rules:

- ▶  $\beta$ -conversions
- ▶  $\eta$ -conversions

$$\frac{\Gamma \vdash t : A \rightarrow B \quad x \text{ does not occur in } t}{\Gamma \vdash t =_{\beta\eta} (\lambda x : A. t x) : A \rightarrow B}$$

$$\frac{\Gamma \vdash t : A \times B}{\Gamma \vdash t =_{\beta\eta} (\text{fst } t, \text{snd } t) : A \times B}$$

$$\frac{\Gamma \vdash t : \text{unit}}{\Gamma \vdash t =_{\beta\eta} () : \text{unit}}$$

# STLC $\beta\eta$ -Equality

The relation  $\Gamma \vdash s =_{\beta\eta} t : A$  (where  $\Gamma$  ranges over typing environments,  $s$  and  $t$  over terms and  $A$  over types) is inductively defined by the following rules:

- ▶  $\beta$ -conversions
- ▶  $\eta$ -conversions
- ▶ congruence rules

$$\frac{\Gamma, x : A \vdash t =_{\beta\eta} t' : B}{\Gamma \vdash \lambda x : A. t =_{\beta\eta} \lambda x : A. t' : A \rightarrow B}$$

$$\frac{\Gamma \vdash s =_{\beta\eta} s' : A \rightarrow B \quad \Gamma \vdash t =_{\beta\eta} t' : A}{\Gamma \vdash st =_{\beta\eta} s't' : B}$$

etc

# STLC $\beta\eta$ -Equality

The relation  $\Gamma \vdash s =_{\beta\eta} t : A$  (where  $\Gamma$  ranges over typing environments,  $s$  and  $t$  over terms and  $A$  over types) is inductively defined by the following rules:

- ▶  $\beta$ -conversions
- ▶  $\eta$ -conversions
- ▶ congruence rules
- ▶  $=_{\beta\eta}$  is reflexive, symmetric and transitive

$\frac{\Gamma \vdash t : A}{\Gamma \vdash t =_{\beta\eta} t : A}$	$\frac{\Gamma \vdash s =_{\beta\eta} t : A}{\Gamma \vdash t =_{\beta\eta} s : A}$
$\frac{\Gamma \vdash r =_{\beta\eta} s : A \quad \Gamma \vdash s =_{\beta\eta} t : A}{\Gamma \vdash r =_{\beta\eta} t : A}$	

# STLC $\beta\eta$ -Equality

**Soundness Theorem** for semantics of STLC in a ccc.

If  $\Gamma \vdash s =_{\beta\eta} t : A$  is provable, then in any ccc

$$M[\Gamma \vdash s : A] = M[\Gamma \vdash t : A]$$

are equal **C**-morphisms  $M[\Gamma] \rightarrow M[A]$ .

**Proof** is by induction on the structure of the proof of  $\Gamma \vdash s =_{\beta\eta} t : A$ .

Here we just check the case of  $\beta$ -conversion for functions.

So suppose we have  $\Gamma, x : A \vdash t : B$  and  $\Gamma \vdash s : A$ . We have to see that

$$M[\Gamma \vdash (\lambda x : A. t)s : B] = M[\Gamma \vdash t[s/x] : B]$$

Suppose

$$M[\Gamma] = X$$

$$M[A] = Y$$

$$M[B] = Z$$

$$M[\Gamma, x : A \vdash t : B] = f : X \times Y \rightarrow Z$$

$$M[\Gamma \vdash s : A] = g : X \rightarrow Z$$

Then

$$M[\Gamma \vdash \lambda x : A. t : A \rightarrow B] = \text{cur } f : X \rightarrow Z^Y$$

and hence

$$M[\Gamma \vdash (\lambda x : A. t)s : B]$$

$$= \text{app} \circ \langle \text{cur } f, g \rangle$$

$$= \text{app} \circ (\text{cur } f \times \text{id}_Y) \circ \langle \text{id}_X, g \rangle$$

$$= f \circ \langle \text{id}_X, g \rangle$$

$$= M[\Gamma \vdash t[s/x] : B]$$

$$\text{since } (a \times b) \circ \langle c, d \rangle = \langle a \circ c, b \circ d \rangle$$

by definition of  $\text{cur } f$

by the Substitution Theorem

as required.

# The internal language of a ccc, $\mathbf{C}$

- ▶ one ground type for each  $\mathbf{C}$ -object  $X$
- ▶ for each  $X \in \mathbf{C}$ , one constant  $f^X$  for each  $\mathbf{C}$ -morphism  $f : 1 \rightarrow X$  (“global element” of the object  $X$ )

The types and terms of STLC over this language usefully describe constructions on the objects and morphisms of  $\mathbf{C}$  using its cartesian closed structure, but in an “element-theoretic” way.

For example...

# Example

In any ccc  $\mathbf{C}$ , for any  $X, Y, Z \in \mathbf{C}$  there is an isomorphism

$$Z^{(X \times Y)} \cong (Z^Y)^X$$



# Example

In any ccc  $\mathbf{C}$ , for any  $X, Y, Z \in \mathbf{C}$  there is an isomorphism

$$Z^{(X \times Y)} \cong (Z^Y)^X$$

which in the internal language of  $\mathbf{C}$  is described by the terms

$$\diamond \vdash s : ((X \times Y) \rightarrow Z) \rightarrow (X \rightarrow (Y \rightarrow Z))$$

$$\diamond \vdash t : (X \rightarrow (Y \rightarrow Z)) \rightarrow ((X \times Y) \rightarrow Z)$$

where  $\begin{cases} s & \triangleq \lambda f : (X \times Y) \rightarrow Z. \lambda x : X. \lambda y : Y. f(x, y) \\ t & \triangleq \lambda g : X \rightarrow (Y \rightarrow Z). \lambda z : X \times Y. g(\text{fst } z) (\text{snd } z) \end{cases}$  and

which satisfy  $\begin{cases} \diamond, f : (X \times Y) \rightarrow Z \vdash t(sf) =_{\beta\eta} f \\ \diamond, g : X \rightarrow (Y \rightarrow Z) \vdash s(tg) =_{\beta\eta} g \end{cases}$

# Free cartesian closed categories

The Soundness Theorem has a converse—completeness.

In fact for a given set of ground types and typed constants there is a single ccc F (the **free ccc** for that language) with an interpretation function  $M$  so that  $\Gamma \vdash s =_{\beta\eta} t : A$  is provable iff  $M[\Gamma \vdash s : A] = M[\Gamma \vdash t : A]$  in **F**.

# Free cartesian closed categories

The Soundness Theorem has a converse—completeness.

In fact for a given set of ground types and typed constants there is a single ccc  $\mathbf{F}$  (the **free ccc** for that language) with an interpretation function  $M$  so that  $\Gamma \vdash s =_{\beta\eta} t : A$  is provable iff  $M[\Gamma \vdash s : A] = M[\Gamma \vdash t : A]$  in  $\mathbf{F}$ .

- ▶  $\mathbf{F}$ -objects are the STLC types over the given set of ground types
- ▶  $\mathbf{F}$ -morphisms  $A \rightarrow B$  are equivalence classes of STLC terms  $t$  satisfying  $\diamond \vdash t : A \rightarrow B$  (so  $t$  is a *closed* term—it has no free variables) with respect to the equivalence relation equating  $s$  and  $t$  if  $\diamond \vdash s =_{\beta\eta} t : A \rightarrow B$  is provable.
- ▶ identity morphism on  $A$  is the equivalence class of  $\diamond \vdash \lambda x : A. x : A \rightarrow A$ .
- ▶ composition of a morphism  $A \rightarrow B$  represented by  $\diamond \vdash s : A \rightarrow B$  and a morphism  $B \rightarrow C$  represented by  $\diamond \vdash t : B \rightarrow C$  is represented by  $\diamond \vdash \lambda x : A. t(s x) : A \rightarrow C$ .

# Curry-Howard correspondence

<b>Logic</b>		<b>Type Theory</b>
propositions	$\leftrightarrow$	types
proofs	$\leftrightarrow$	terms

E.g. IPL *versus* STLC.

# Curry-Howard for IPL vs STLC

Proof of  $\diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta \vdash \varphi \Rightarrow \theta$  in IPL

$$\frac{\frac{\frac{\dots (AX)}{\Phi \vdash \psi \Rightarrow \theta} (WK) \quad \frac{\frac{\frac{\dots (AX)}{\Phi \vdash \varphi \Rightarrow \psi} (WK)}{\Phi \vdash \psi} (\Rightarrow E)}{\Phi \vdash \theta} (\Rightarrow E)}{\diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta \vdash \varphi \Rightarrow \theta} (\Rightarrow I)}{\Phi \vdash \varphi} (AX) (\Rightarrow E)$$

where  $\Phi = \diamond, \varphi \Rightarrow \psi, \psi \Rightarrow \theta, \varphi$

# Curry-Howard for IPL vs STLC

and a corresponding STLC term

$$\frac{\frac{\frac{\dots (AX)}{\Phi \vdash z : \psi \Rightarrow \theta} (WK) \quad \frac{\frac{\frac{\dots (AX)}{\dots} (WK) \quad \Phi \vdash y : \varphi \Rightarrow \psi} (WK) \quad \Phi \vdash x : \varphi} (\Rightarrow E)}{\Phi \vdash z(yx) : \theta} (\Rightarrow E)}{\diamond, y : \varphi \Rightarrow \psi, z : \psi \Rightarrow \theta \vdash \lambda x : \varphi. z(yx) : \varphi \Rightarrow \theta} (\Rightarrow I)$$

where  $\Phi = \diamond, y : \varphi \Rightarrow \psi, z : \psi \Rightarrow \theta, x : \varphi$

# Curry-Howard-Lawvere/Lambek correspondence

<b>Logic</b>		<b>Type Theory</b>		<b>Category Theory</b>
propositions	$\leftrightarrow$	types	$\leftrightarrow$	objects
proofs	$\leftrightarrow$	terms	$\leftrightarrow$	morphisms

E.g. IPL *versus* STLC *versus* CCCs

# Curry-Howard-Lawvere/Lambek correspondence

<b>Logic</b>		<b>Type Theory</b>		<b>Category Theory</b>
propositions	$\leftrightarrow$	types	$\leftrightarrow$	objects
proofs	$\leftrightarrow$	terms	$\leftrightarrow$	morphisms

E.g. IPL *versus* STLC *versus* CCCs

These correspondences can be made into category-theoretic equivalences—we first need to define the notions of **functor** and **natural transformation** in order to define the notion of **equivalence of categories**.



# Lecture 10

# Functors

are the appropriate notion of morphism between categories

Given categories  $\mathbf{C}$  and  $\mathbf{D}$ , a functor  $F : \mathbf{C} \rightarrow \mathbf{D}$  is specified by:

- ▶ a function  $\text{obj } \mathbf{C} \rightarrow \text{obj } \mathbf{D}$  whose value at  $X$  is written  $FX$
- ▶ for all  $X, Y \in \mathbf{C}$ , a function  $\mathbf{C}(X, Y) \rightarrow \mathbf{D}(FX, FY)$  whose value at  $f : X \rightarrow Y$  is written  $Ff : FX \rightarrow FY$

and which is required to preserve composition and identity morphisms:

$$F(g \circ f) = Fg \circ Ff$$

$$F(\text{id}_X) = \text{id}_{FX}$$

# Examples of functors

“Forgetful” functors from categories of set-with-structure back to **Set**.

E.g.  $U : \mathbf{Mon} \rightarrow \mathbf{Set}$

$$\begin{cases} U(M, \cdot, e) & = M \\ U((M_1, \cdot_1, e_1) \xrightarrow{f} (M_2, \cdot_2, e_2)) & = M_1 \xrightarrow{f} M_2 \end{cases}$$

# Examples of functors

“Forgetful” functors from categories of set-with-structure back to **Set**.

E.g.  $U : \mathbf{Mon} \rightarrow \mathbf{Set}$

$$\begin{cases} U(M, \cdot, e) & = M \\ U((M_1, \cdot_1, e_1) \xrightarrow{f} (M_2, \cdot_2, e_2)) & = M_1 \xrightarrow{f} M_2 \end{cases}$$

Similarly  $U : \mathbf{Preord} \rightarrow \mathbf{Set}$ .

# Examples of functors

Free monoid functor  $F : \text{Set} \rightarrow \text{Mon}$

Given  $\Sigma \in \text{Set}$ ,

$F \Sigma = (\text{List } \Sigma, @, \text{nil})$ , the free monoid on  $\Sigma$

# Examples of functors

Free monoid functor  $F : \text{Set} \rightarrow \text{Mon}$

Given  $\Sigma \in \text{Set}$ ,

$F \Sigma = (\text{List } \Sigma, @, \text{nil})$ , the free monoid on  $\Sigma$

Given a function  $f : \Sigma_1 \rightarrow \Sigma_2$ , we get a function

$F f : \text{List } \Sigma_1 \rightarrow \text{List } \Sigma_2$  by **mapping**  $f$  over finite lists:

$$F f [a_1, \dots, a_n] = [f a_1, \dots, f a_n]$$

This gives a monoid morphism  $F \Sigma_1 \rightarrow F \Sigma_2$ ; and mapping over lists preserves composition ( $F(g \circ f) = F g \circ F f$ ) and identities ( $F \text{id}_\Sigma = \text{id}_{F \Sigma}$ ). So we do get a functor from **Set** to **Mon**.

# Examples of functors

If  $\mathbf{C}$  is a category with binary products and  $X \in \mathbf{C}$ , then the function  $(\_) \times X : \text{obj } \mathbf{C} \rightarrow \text{obj } \mathbf{C}$  extends to a functor  $(\_) \times X : \mathbf{C} \rightarrow \mathbf{C}$  mapping morphisms  $f : Y \rightarrow Y'$  to

$$f \times \text{id}_X : Y \times X \rightarrow Y' \times X$$

(recall that  $f \times g$  is the unique morphism with  $\begin{cases} \pi_1 \circ (f \times g) &= f \circ \pi_1 \\ \pi_2 \circ (f \times g) &= g \circ \pi_2 \end{cases}$ )

since it is the case that

$$\begin{cases} \text{id}_X \times \text{id}_Y &= \text{id}_{X \times Y} \\ (f' \circ f) \times \text{id}_X &= (f' \times \text{id}_X) \circ (f \times \text{id}_X) \end{cases}$$

(see Exercise Sheet 2, question 1c).

# Examples of functors

If  $\mathbf{C}$  is a cartesian closed category and  $X \in \mathbf{C}$ , then the function  $(\_)^X : \text{obj } \mathbf{C} \rightarrow \text{obj } \mathbf{C}$  extends to a functor

$(\_)^X : \mathbf{C} \rightarrow \mathbf{C}$  mapping morphisms  $f : Y \rightarrow Y'$  to

$$f^X \triangleq \text{cur}(f \circ \text{app}) : Y^X \rightarrow Y'^X$$

since it is the case that 
$$\begin{cases} (\text{id}_Y)^X & = \text{id}_{Y^X} \\ (g \circ f)^X & = g^X \circ f^X \end{cases}$$

(see Exercise Sheet 3, question 4).



# Contravariance

Given categories  $\mathbf{C}$  and  $\mathbf{D}$ , a functor  $F : \mathbf{C}^{\text{op}} \rightarrow \mathbf{D}$  is called a **contravariant functor from  $\mathbf{C}$  to  $\mathbf{D}$** .

Note that if  $X \xrightarrow{f} Y \xrightarrow{g} Z$  in  $\mathbf{C}$ , then  $X \xleftarrow{f} Y \xleftarrow{g} Z$  in  $\mathbf{C}^{\text{op}}$

so  $F X \xleftarrow{Ff} F Y \xleftarrow{Fg} F Z$  in  $\mathbf{D}$  and hence

$$F(g \circ_{\mathbf{C}} f) = F f \circ_{\mathbf{D}} F g$$

(contravariant functors **reverse the order of composition**)

A functor  $\mathbf{C} \rightarrow \mathbf{D}$  is sometimes called a **covariant functor from  $\mathbf{C}$  to  $\mathbf{D}$** .

# Example of a contravariant functor

If  $\mathbf{C}$  is a cartesian closed category and  $X \in \mathbf{C}$ , then the function  $X^{(-)} : \text{obj } \mathbf{C} \rightarrow \text{obj } \mathbf{C}$  extends to a functor

$X^{(-)} : \mathbf{C}^{\text{op}} \rightarrow \mathbf{C}$  mapping morphisms  $f : Y \rightarrow Y'$  to

$$X^f \triangleq \text{cur}(\text{app} \circ (\text{id}_{X^{Y'}} \times f)) : X^{Y'} \rightarrow X^Y$$

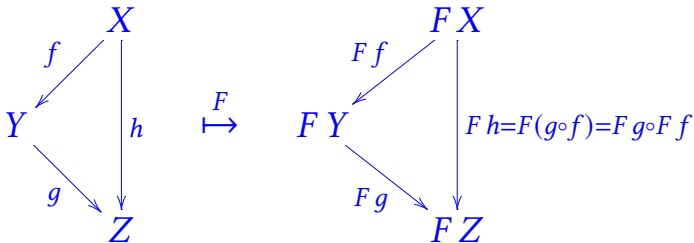
since it is the case that 
$$\begin{cases} X^{\text{id}_Y} & = \text{id}_{X^Y} \\ X^{g \circ f} & = X^f \circ X^g \end{cases}$$

(see Exercise Sheet 3, question 5).

Note that since a functor  $F : \mathbf{C} \rightarrow \mathbf{D}$  preserves domains, codomains, composition and identity morphisms

it sends commutative diagrams in  $\mathbf{C}$  to commutative diagrams in  $\mathbf{D}$

E.g.



Note that since a functor  $F : \mathbf{C} \rightarrow \mathbf{D}$  preserves domains, codomains, composition and identity morphisms **it sends isomorphisms in  $\mathbf{C}$  to isomorphisms in  $\mathbf{D}$** , because

$$\begin{array}{ccc}
 X & \xrightarrow{f} & Y \\
 \searrow \text{id}_X & & \downarrow g \\
 & & X \xrightarrow{f} Y \\
 & & \swarrow \text{id}_Y
 \end{array}
 \xrightarrow{F}
 \begin{array}{ccc}
 FX & \xrightarrow{Ff} & FY \\
 \searrow \text{id}_{FX} & & \downarrow Fg \\
 & & FX \xrightarrow{Ff} FY \\
 & & \swarrow \text{id}_{FY}
 \end{array}$$

so  $F(f^{-1}) = (Ff)^{-1}$

# Composing functors

Given functors  $F : \mathbf{C} \rightarrow \mathbf{D}$  and  $G : \mathbf{D} \rightarrow \mathbf{E}$ , we get a functor  $G \circ F : \mathbf{C} \rightarrow \mathbf{E}$  with

$$G \circ F \left( \begin{array}{c} X \\ \downarrow f \\ Y \end{array} \right) = \begin{array}{c} G(F X) \\ \downarrow G(F f) \\ G(F Y) \end{array}$$

(this preserves composition and identity morphisms, because  $F$  and  $G$  do)

# Identity functor

on a category  $\mathbf{C}$  is  $\text{id}_{\mathbf{C}} : \mathbf{C} \rightarrow \mathbf{C}$  where

$$\text{id}_{\mathbf{C}} \left( \begin{array}{c} X \\ \downarrow f \\ Y \end{array} \right) = \begin{array}{c} X \\ \downarrow f \\ Y \end{array}$$

Functor composition and identity functors satisfy

**associativity**       $H \circ (G \circ F) = (H \circ G) \circ F$

**unity**               $\text{id}_D \circ F = F = F \circ \text{id}_C$

So we can get categories whose objects are categories  
and whose morphisms are functors

but we have to be a bit careful about **size**...

# Size

One of the axioms of set theory is

**set membership is a well-founded relation**, that is, there is no infinite sequence of sets  $X_0, X_1, X_2, \dots$  with

$$\dots \in X_{n+1} \in X_n \in \dots \in X_2 \in X_1 \in X_0$$

So in particular there is no set  $X$  with  $X \in X$ .

So we cannot form the “set of all sets” or the “category of all categories”.



# Size

One of the axioms of set theory is

**set membership is a well-founded relation**, that is, there is no infinite sequence of sets  $X_0, X_1, X_2, \dots$  with

$$\dots \in X_{n+1} \in X_n \in \dots \in X_2 \in X_1 \in X_0$$

So in particular there is no set  $X$  with  $X \in X$ .

So we cannot form the “set of all sets” or the “category of all categories”.

But we do assume there are (lots of) big sets

$$\mathcal{U}_0 \in \mathcal{U}_1 \in \mathcal{U}_2 \in \dots$$

where “big” means each  $\mathcal{U}_n$  is a **Grothendieck universe**...

# Grothendieck universes

A **Grothendieck universe**  $\mathcal{U}$  is a set of sets satisfying

- ▶  $X \in Y \in \mathcal{U} \Rightarrow X \in \mathcal{U}$
- ▶  $X, Y \in \mathcal{U} \Rightarrow \{X, Y\} \in \mathcal{U}$
- ▶  $X \in \mathcal{U} \Rightarrow \mathcal{P}X \triangleq \{Y \mid Y \subseteq X\} \in \mathcal{U}$
- ▶  $X \in \mathcal{U} \wedge F \in \mathcal{U}^X \Rightarrow$   
 $\{y \mid \exists x \in X, y \in Fx\} \in \mathcal{U}$   
(hence also  $X, Y \in \mathcal{U} \Rightarrow X \times Y \in \mathcal{U} \wedge Y^X \in \mathcal{U}$ )

The above properties are satisfied by  $\mathcal{U} = \emptyset$ , but we will always assume

- ▶  $\mathbb{N} \in \mathcal{U}$

# Size

We assume

there is an infinite sequence  $\mathcal{U}_0 \in \mathcal{U}_1 \in \mathcal{U}_2 \in \dots$  of bigger and bigger Grothendieck universes

and revise the previous definition of “the” category of sets and functions:

$\mathbf{Set}_n$  = category whose objects are all the sets in  $\mathcal{U}_n$  and with  $\mathbf{Set}_n(X, Y) = Y^X =$  all functions from  $X$  to  $Y$ .

**Notation:**  $\mathbf{Set} \triangleq \mathbf{Set}_0$  — its objects are called **small sets** (and other sets we call **large**).

# Size

**Set** is the category of small sets.

**Definition.** A category  $\mathbf{C}$  is **locally small** if for all  $X, Y \in \mathbf{C}$ , the set of  $\mathbf{C}$ -morphisms  $X \rightarrow Y$  is small, that is,  $\mathbf{C}(X, Y) \in \mathbf{Set}$ .

$\mathbf{C}$  is a **small category** if it is both locally small and  $\text{obj } \mathbf{C} \in \mathbf{Set}$ .

E.g. **Set**, **Preord** and **Mon** are all locally small (but not small).

Given  $P \in \mathbf{Preord}$ , the category  $\mathbf{C}_P$  it determines is small; similarly, the category  $\mathbf{C}_M$  determined by  $M \in \mathbf{Mon}$  is small.

# The category of small categories, $\mathbf{Cat}$

- ▶ objects are all small categories
- ▶ morphisms in  $\mathbf{Cat}(\mathbf{C}, \mathbf{D})$  are all functors  $\mathbf{C} \rightarrow \mathbf{D}$
- ▶ composition and identity morphisms as for functors

$\mathbf{Cat}$  is a locally small category

# Lecture 11

# The category of small categories

Recall definition of **Cat**:

- ▶ objects are all small categories
- ▶ morphisms in **Cat(C, D)** are all functors  $C \rightarrow D$
- ▶ composition and identity morphisms as for functors

# Cat has a terminal object

The category

$$0 \xrightarrow{\text{id}_0} 0$$

one object, one morphism

is terminal in **Cat**



# Cat has binary products

Given small categories  $\mathbf{C}, \mathbf{D} \in \mathbf{Cat}$ , their product

$\mathbf{C} \xleftarrow{\pi_1} \mathbf{C} \times \mathbf{D} \xrightarrow{\pi_2} \mathbf{D}$  is:

# Cat has binary products

Given small categories  $\mathbf{C}, \mathbf{D} \in \mathbf{Cat}$ , their product

$\mathbf{C} \xleftarrow{\pi_1} \mathbf{C} \times \mathbf{D} \xrightarrow{\pi_2} \mathbf{D}$  is:

- ▶ objects of  $\mathbf{C} \times \mathbf{D}$  are pairs  $(X, Y)$  where  $X \in \mathbf{C}$  and  $Y \in \mathbf{D}$
- ▶ morphisms  $(X, Y) \rightarrow (X', Y')$  in  $\mathbf{C} \times \mathbf{D}$  are pairs  $(f, g)$  where  $f \in \mathbf{C}(X, X')$  and  $g \in \mathbf{D}(Y, Y')$
- ▶ composition and identity morphisms are given by those of  $\mathbf{C}$  (in the first component) and  $\mathbf{D}$  (in the second component)

$$\left\{ \begin{array}{l} \pi_1 \left( (X, Y) \xrightarrow{(f, g)} (X', Y') \right) = X \xrightarrow{f} X' \\ \pi_2 \left( (X, Y) \xrightarrow{(f, g)} (X', Y') \right) = Y \xrightarrow{g} Y' \end{array} \right.$$

**Cat** not only has finite products, it is also cartesian closed.

Exponentials in **Cat** are called **functor categories**.

To define them we need to consider **natural transformations**, which are the appropriate notion of morphism between functors.

# Natural transformations

**Motivating example:** fix a set  $S \in \mathbf{Set}$  and consider the two functors  $F, G : \mathbf{Set} \rightarrow \mathbf{Set}$  given by

$$F \left( X \xrightarrow{f} Y \right) = S \times X \xrightarrow{\text{id}_S \times f} S \times Y$$

$$G \left( X \xrightarrow{f} Y \right) = X \times S \xrightarrow{f \times \text{id}_S} Y \times S$$

# Natural transformations

**Motivating example:** fix a set  $S \in \mathbf{Set}$  and consider the two functors  $F, G : \mathbf{Set} \rightarrow \mathbf{Set}$  given by

$$F \left( X \xrightarrow{f} Y \right) = S \times X \xrightarrow{\text{id}_S \times f} S \times Y$$
$$G \left( X \xrightarrow{f} Y \right) = X \times S \xrightarrow{f \times \text{id}_S} Y \times S$$

For each  $X \in \mathbf{Set}$  there is an isomorphism (bijection)  $\theta_X : FX \cong GX$  in  $\mathbf{Set}$  given by  $\langle \pi_2, \pi_1 \rangle : S \times X \rightarrow X \times S$ .

These isomorphisms do not depend on the particular nature of each set  $X$  (they are “polymorphic in  $X$ ”). One way to make this precise is...

...if we change from  $X$  to  $Y$  along a function  $f : X \rightarrow Y$ , then we get a commutative diagram in **Set**:

$$\begin{array}{ccc} S \times X & \xrightarrow{\langle \pi_2, \pi_1 \rangle} & X \times S \\ \text{id} \times f \downarrow & & \downarrow f \times \text{id} \\ S \times Y & \xrightarrow{\langle \pi_2, \pi_1 \rangle} & Y \times S \end{array}$$

The square commutes because for all  $s \in S$  and  $x \in X$

$$\begin{aligned} \langle \pi_2, \pi_1 \rangle((\text{id} \times f)(s, x)) &= \langle \pi_2, \pi_1 \rangle(s, f x) \\ &= (f x, s) \\ &= (f \times \text{id})(x, s) \\ &= (f \times \text{id})(\langle \pi_2, \pi_1 \rangle(s, x)) \end{aligned}$$

...if we change from  $X$  to  $Y$  along a function  $f : X \rightarrow Y$ , then we get a commutative diagram in **Set**:

$$\begin{array}{ccc} F X & \xrightarrow{\theta_X} & G X \\ F f \downarrow & & \downarrow G f \\ F Y & \xrightarrow{\theta_Y} & G Y \end{array}$$

We say that the family  $(\theta_X \mid X \in \mathbf{Set})$  is **natural** in  $X$ .

# Natural transformations

**Definition.** Given categories and functors  $F, G : \mathbf{C} \rightarrow \mathbf{D}$ , a **natural transformation**  $\theta : F \rightarrow G$  is a family of  $\mathbf{D}$ -morphisms  $\theta_X \in \mathbf{D}(F X, G X)$ , one for each  $X \in \mathbf{C}$ , such that for all  $\mathbf{C}$ -morphisms  $f : X \rightarrow Y$ , the diagram

$$\begin{array}{ccc} F X & \xrightarrow{\theta_X} & G X \\ F f \downarrow & & \downarrow G f \\ F Y & \xrightarrow{\theta_Y} & G Y \end{array}$$

commutes in  $\mathbf{D}$ , that is,  $\theta_Y \circ F f = G f \circ \theta_X$ .



# Example

Recall forgetful ( $U$ ) and free ( $F$ ) functors:

$$\text{Set} \begin{array}{c} \xleftarrow{U} \\ \xrightarrow{F} \end{array} \text{Mon}$$

There is a natural transformation  $\eta : \text{id}_{\text{Set}} \rightarrow U \circ F$ , where for each  $\Sigma \in \text{Set}$

$$\eta_{\Sigma} : \Sigma \rightarrow U(F \Sigma) = \text{List } \Sigma$$

$$a \in \Sigma \mapsto [a] \in \text{List } \Sigma \text{ (one-element list)}$$

(Easy to see that

$$\begin{array}{ccc} \Sigma & \xrightarrow{\eta_{\Sigma}} & U(F \Sigma) \\ f \downarrow & & \downarrow U(F f) \\ \Sigma' & \xrightarrow{\eta_{\Sigma'}} & U(F \Sigma') \end{array} \text{ commutes.)}$$

# Example

The **covariant powerset functor**  $\mathcal{P} : \mathbf{Set} \rightarrow \mathbf{Set}$  is

$$\begin{aligned}\mathcal{P} X &\triangleq \{S \mid S \subseteq X\} \\ \mathcal{P} \left( X \xrightarrow{f} Y \right) &\triangleq \mathcal{P} X \xrightarrow{\mathcal{P} f} \mathcal{P} Y \\ S &\mapsto \mathcal{P} f S \triangleq \{f x \mid x \in S\}\end{aligned}$$

# Example

The **covariant powerset functor**  $\mathcal{P} : \mathbf{Set} \rightarrow \mathbf{Set}$  is

$$\begin{aligned}\mathcal{P} X &\triangleq \{S \mid S \subseteq X\} \\ \mathcal{P} \left( X \xrightarrow{f} Y \right) &\triangleq \mathcal{P} X \xrightarrow{\mathcal{P} f} \mathcal{P} Y \\ S &\mapsto \mathcal{P} f S \triangleq \{f x \mid x \in S\}\end{aligned}$$

There is a natural transformation  $\cup : \mathcal{P} \circ \mathcal{P} \rightarrow \mathcal{P}$  whose component at  $X \in \mathbf{Set}$  sends  $\mathcal{S} \in \mathcal{P}(\mathcal{P} X)$  to

$$\cup_X \mathcal{S} \triangleq \{x \in X \mid \exists S \in \mathcal{S}, x \in S\} \in \mathcal{P} X$$

(check that  $\cup_X$  is natural in  $X$ )

# Non-example

The classic example of an “un-natural transformation” (the one that caused Eilenburg and MacLane to invent the concept of naturality) is the linear isomorphism between a finite dimensional real vectorspace  $V$  and its dual  $V^*$  (= vectorspace of linear functions  $V \rightarrow \mathbb{R}$ ).

Both  $V$  and  $V^*$  have the same finite dimension, so are isomorphic by choosing bases; but there is no choice of basis for each  $V$  that makes the family of isomorphisms natural in  $V$ .

For a similar, more elementary non-example, see Ex. Sh. 5, question 4.

# Composing natural transformations

Given functors  $F, G, H : \mathbf{C} \rightarrow \mathbf{D}$  and natural transformations  $\theta : F \rightarrow G$  and  $\varphi : G \rightarrow H$ ,

we get  $\varphi \circ \theta : F \rightarrow H$  with

$$(\varphi \circ \theta)_X = \left( F X \xrightarrow{\theta_X} G X \xrightarrow{\varphi_X} H X \right)$$

# Composing natural transformations

Given functors  $F, G, H : \mathbf{C} \rightarrow \mathbf{D}$  and natural transformations  $\theta : F \rightarrow G$  and  $\varphi : G \rightarrow H$ ,

we get  $\varphi \circ \theta : F \rightarrow H$  with

$$(\varphi \circ \theta)_X = \left( F X \xrightarrow{\theta_X} G X \xrightarrow{\varphi_X} H X \right)$$

Check naturality:

$$\begin{aligned} H f \circ (\varphi \circ \theta)_X &\triangleq H f \circ \varphi_X \circ \theta_X \\ &= \varphi_Y \circ G f \circ \theta_X && \text{naturality of } \varphi \\ &= \varphi_Y \circ \theta_Y \circ F f \\ &\triangleq (\varphi \circ \theta)_Y \circ F f && \text{naturality of } \theta \end{aligned}$$

# Identity natural transformation

Given a functor  $F : \mathbf{C} \rightarrow \mathbf{D}$ , we get a natural transformation  $\text{id}_F : F \rightarrow F$  with

$$(\text{id}_F)_X = F X \xrightarrow{\text{id}_{FX}} F X$$

# Identity natural transformation

Given a functor  $F : \mathbf{C} \rightarrow \mathbf{D}$ , we get a natural transformation  $\text{id}_F : F \rightarrow F$  with

$$(\text{id}_F)_X = F X \xrightarrow{\text{id}_{FX}} F X$$

Check naturality:

$$F f \circ (\text{id}_F)_X \triangleq F f \circ \text{id}_{FX} = F f = \text{id}_{FY} \circ F f \triangleq (\text{id}_F)_Y \circ F f$$



# Functor categories

It is easy to see that composition and identities for natural transformations satisfy

$$(\psi \circ \varphi) \circ \theta = \psi \circ (\varphi \circ \theta)$$

$$\text{id}_G \circ \theta = \theta \circ \text{id}_F$$

so that we get a category:

**Definition.** Given categories  $\mathbf{C}$  and  $\mathbf{D}$ , the **functor category**  $\mathbf{D}^{\mathbf{C}}$  has

- ▶ objects are all functors  $\mathbf{C} \rightarrow \mathbf{D}$
- ▶ given  $F, G : \mathbf{C} \rightarrow \mathbf{D}$ , morphism from  $F$  to  $G$  in  $\mathbf{D}^{\mathbf{C}}$  are the natural transformations  $F \rightarrow G$
- ▶ composition and identity morphisms as above

If  $\mathcal{U}$  is a Grothendieck universe, then for each  $X \in \mathcal{U}$  and  $F \in \mathcal{U}^X$  we have that their **dependent product** and **dependent function** sets

$$\sum_{x \in X} F x \triangleq \{(x, y) \mid x \in X \wedge y \in F x\}$$

$$\prod_{x \in X} F x \triangleq \{f \subseteq \sum_{x \in X} F x \mid f \text{ is single-valued and total}\}$$

are also in  $\mathcal{U}$ ; and as a special case (of  $\prod$ , when  $F$  is a constant function with value  $Y$ ) we also have that  $X, Y \in \mathcal{U}$  implies  $Y^X \in \mathcal{U}$ .

If  $\mathcal{U}$  is a Grothendieck universe, then for each  $X \in \mathcal{U}$  and  $F \in \mathcal{U}^X$  we have that their **dependent product** and **dependent function** sets

$$\sum_{x \in X} F x \triangleq \{(x, y) \mid x \in X \wedge y \in F x\}$$

$$\prod_{x \in X} F x \triangleq \{f \subseteq \sum_{x \in X} F x \mid f \text{ is single-valued and total}\}$$

are also in  $\mathcal{U}$ ; and as a special case (of  $\prod$ , when  $F$  is a constant function with value  $Y$ ) we also have that  $X, Y \in \mathcal{U}$  implies  $Y^X \in \mathcal{U}$ . Hence

If  $\mathbf{C}$  and  $\mathbf{D}$  are small categories, then so is  $\mathbf{D}^{\mathbf{C}}$ .

because

$$\begin{aligned} \text{obj}(\mathbf{D}^{\mathbf{C}}) &\subseteq \sum_{F \in (\text{obj } \mathbf{D})^{\text{obj } \mathbf{C}}} \prod_{X, Y \in \text{obj } \mathbf{C}} \mathbf{D}(F X, F Y)^{\mathbf{C}(X, Y)} \\ \mathbf{D}^{\mathbf{C}}(F, G) &\subseteq \prod_{X \in \text{obj } \mathbf{C}} \mathbf{D}(F X, G X) \end{aligned}$$

If  $\mathcal{U}$  is a Grothendieck universe, then for each  $X \in \mathcal{U}$  and  $F \in \mathcal{U}^X$  we have that their **dependent product** and **dependent function** sets

$$\sum_{x \in X} F x \triangleq \{(x, y) \mid x \in X \wedge y \in F x\}$$

$$\prod_{x \in X} F x \triangleq \{f \subseteq \sum_{x \in X} F x \mid f \text{ is single-valued and total}\}$$

are also in  $\mathcal{U}$ ; and as a special case (of  $\prod$ , when  $F$  is a constant function with value  $Y$ ) we also have that  $X, Y \in \mathcal{U}$  implies  $Y^X \in \mathcal{U}$ . Hence

If  $\mathbf{C}$  and  $\mathbf{D}$  are small categories, then so is  $\mathbf{D}^{\mathbf{C}}$ .

because

$$\text{obj}(\mathbf{D}^{\mathbf{C}}) \subseteq \sum_{F \in (\text{obj } \mathbf{D})^{\text{obj } \mathbf{C}}} \prod_{X, Y \in \text{obj } \mathbf{C}} \mathbf{D}(F X, F Y)^{\mathbf{C}(X, Y)}$$

$$\mathbf{D}^{\mathbf{C}}(F, G) \subseteq \prod_{X \in \text{obj } \mathbf{C}} \mathbf{D}(F X, G X)$$

**Aim** to show that functor category  $\mathbf{D}^{\mathbf{C}}$  is the exponential of  $\mathbf{C}$  and  $\mathbf{D}$  in  $\mathbf{Cat}$ ...

# Cat is cartesian closed

**Theorem.** There is an **application functor**

$$\text{app} : \mathbf{D}^{\mathbf{C}} \times \mathbf{C} \rightarrow \mathbf{D}$$

that makes  $\mathbf{D}^{\mathbf{C}}$  the exponential for  $\mathbf{C}$  and  $\mathbf{D}$  in **Cat**.

Given  $(F, X) \in \mathbf{D}^{\mathbf{C}} \times \mathbf{C}$ , we define

$$\text{app}(F, X) \triangleq FX$$

and given  $(\theta, f) : (F, X) \rightarrow (G, Y)$  in  $\mathbf{D}^{\mathbf{C}} \times \mathbf{C}$ , we define

$$\begin{aligned} \text{app}\left((F, X) \xrightarrow{(\theta, f)} (G, Y)\right) &\triangleq FX \xrightarrow{Ff} FY \xrightarrow{\theta_Y} GY \\ &= FX \xrightarrow{\theta_X} GX \xrightarrow{Gf} GY \end{aligned}$$

Check: 
$$\begin{cases} \text{app}(\text{id}_F, \text{id}_X) &= \text{id}_{FX} \\ \text{app}(\varphi \circ \theta, g \circ f) &= \text{app}(\varphi, g) \circ \text{app}(\theta, f) \end{cases}$$

# Cat is cartesian closed

**Theorem.** There is an **application functor**

$$\text{app} : \mathbf{D}^{\mathbf{C}} \times \mathbf{C} \rightarrow \mathbf{D}$$

that makes  $\mathbf{D}^{\mathbf{C}}$  the exponential for  $\mathbf{C}$  and  $\mathbf{D}$  in **Cat**.

Definition of currying: given functor  $F : \mathbf{E} \times \mathbf{C} \rightarrow \mathbf{D}$ , we get a functor  $\text{cur } F : \mathbf{E} \rightarrow \mathbf{D}^{\mathbf{C}}$  as follows. For each  $Z \in \mathbf{E}$ ,  $\text{cur } F Z \in \mathbf{D}^{\mathbf{C}}$  is the functor

$$\text{cur } F Z \left( \begin{array}{c} X \\ \downarrow f \\ X' \end{array} \right) \triangleq \begin{array}{c} F(Z, X) \\ \downarrow F(\text{id}_Z, f) \\ F(Z, X') \end{array}$$

For each  $g : Z \rightarrow Z'$  in  $\mathbf{E}$ ,  $\text{cur } F g : \text{cur } F Z \rightarrow \text{cur } F Z'$  is the natural transformation whose component at each  $X \in \mathbf{C}$  is

$$(\text{cur } F g)_X \triangleq F(g, \text{id}_X) : F(Z, X) \rightarrow F(Z', X)$$

(Check that this is natural in  $X$ ; and that  $\text{cur } F$  preserves composition and identities in  $\mathbf{E}$ .)

# Cat is cartesian closed

**Theorem.** There is an **application functor**  
 $\text{app} : \mathbf{D}^{\mathbf{C}} \times \mathbf{C} \rightarrow \mathbf{D}$   
that makes  $\mathbf{D}^{\mathbf{C}}$  the exponential for  $\mathbf{C}$  and  $\mathbf{D}$  in **Cat**.

Have to check that  $\text{cur } F$  is the unique functor  $G : \mathbf{E} \rightarrow \mathbf{D}^{\mathbf{C}}$  that makes

$$\begin{array}{ccc} \mathbf{E} \times \mathbf{C} & \xrightarrow{F} & \mathbf{D} \\ G \times \text{id}_{\mathbf{C}} \downarrow & \nearrow \text{app} & \\ \mathbf{D}^{\mathbf{C}} \times \mathbf{C} & & \end{array}$$

commute in **Cat** (**exercise**).

## Lecture 12



# Adjoint functors

The concepts of “category”, “functor” and “natural transformation” were invented by Eilenberg and MacLane in order to formalise “adjoint situations”.

They appear everywhere in mathematics, logic and (hence) computer science.

Examples of adjoint situations that we have already seen...

# Free monoids

$$\frac{\Sigma \rightarrow U(M, \cdot, e) \text{ morphisms in Set}}{F \Sigma \rightarrow (M, \cdot, e) \text{ morphisms in Mon}}$$

bijection

$$\text{Set}(\Sigma, U(M, \cdot, e)) \cong \text{Mon}(F \Sigma, (M, \cdot, e))$$

$$f \mapsto \hat{f}$$

$$g \circ \eta_\Sigma \leftarrow g$$

(where  $\eta_\Sigma : \Sigma \rightarrow F \Sigma = \text{List } \Sigma$  is  $a \mapsto [a]$ )

The bijection is “natural in  $\Sigma$  and  $(M, \cdot, e)$ ” (to be explained)

# Binary product in a category $\mathbf{C}$

$$\frac{(Z, Z) \rightarrow (X, Y) \text{ morphisms in } \mathbf{C} \times \mathbf{C}}{Z \rightarrow X \times Y \text{ morphisms in } \mathbf{C}}$$

bijection

$$(\mathbf{C} \times \mathbf{C})((Z, Z), (X, Y)) \cong \mathbf{C}(Z, X \times Y)$$

$$(f, g) \mapsto \langle f, g \rangle$$

$$(\pi_1 \circ h, \pi_2 \circ h) \leftarrow h$$

This bijection is “natural in  $X, Y, Z$ ” (to be explained)

# Exponentials in a category $\mathbf{C}$ with binary products

$$\frac{Z \times X \rightarrow Y \text{ morphisms in } \mathbf{C}}{\underline{\underline{Z \rightarrow Y^X \text{ morphisms in } \mathbf{C}}}}$$

bijection

$$\mathbf{C}(Z \times X, Y) \cong \mathbf{C}(Z, Y^X)$$

$$f \mapsto \text{cur } f$$

$$\text{app} \circ (g \times \text{id}_X) \leftarrow g$$

The bijection is “natural in  $X, Y, Z$ ” (to be explained)

# Adjunction

**Definition.** An **adjunction** between two categories **C** and **D** is specified by:

- ▶ functors  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D}$
- ▶ for each  $X \in \mathbf{C}$  and  $Y \in \mathbf{D}$  a bijection  $\theta_{X,Y} : \mathbf{D}(FX, Y) \cong \mathbf{C}(X, GY)$  which is **natural in X and Y**.

for all  $\begin{cases} u : X' \rightarrow X \text{ in } \mathbf{C} \\ v : Y \rightarrow Y' \text{ in } \mathbf{D} \end{cases}$  and all  $g : FX \rightarrow Y$  in  $\mathbf{D}$

$$X' \xrightarrow{u} X \xrightarrow{\theta_{X,Y}(g)} GY \xrightarrow{Gv} GY' = \theta_{X',Y'} \left( FX' \xrightarrow{Fu} FX \xrightarrow{g} Y \xrightarrow{v} Y' \right)$$

# Adjunction

**Definition.** An **adjunction** between two categories **C** and **D** is specified by:

▶ functors  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D}$

▶ for each  $X \in \mathbf{C}$  and  $Y \in \mathbf{D}$  a bijection

$$\theta_{X,Y} : \mathbf{D}(F X, Y) \cong \mathbf{C}(X, G Y)$$

which is **natural in  $X$  and  $Y$** .

what has this to do with the concept of natural transformation between functors?

# Hom functors

If  $\mathbf{C}$  is a locally small category, then we get a functor

$$\text{Hom}_{\mathbf{C}} : \mathbf{C}^{\text{op}} \times \mathbf{C} \rightarrow \mathbf{Set}$$

with  $\text{Hom}_{\mathbf{C}}(X, Y) \triangleq \mathbf{C}(X, Y)$  and

$$\text{Hom}_{\mathbf{C}} \left( (X, Y) \xrightarrow{(f, g)} (X', Y') \right) \triangleq \mathbf{C}(X, Y) \xrightarrow{\text{Hom}_{\mathbf{C}}(f, g)} \mathbf{C}(X', Y')$$
$$\text{Hom}_{\mathbf{C}}(f, g) h \triangleq g \circ h \circ f$$

# Hom functors

If  $\mathbf{C}$  is a locally small category, then we get a functor

$$\text{Hom}_{\mathbf{C}} : \mathbf{C}^{\text{op}} \times \mathbf{C} \rightarrow \mathbf{Set}$$

with  $\text{Hom}_{\mathbf{C}}(X, Y) \triangleq \mathbf{C}(X, Y)$  and

$$\text{Hom}_{\mathbf{C}} \left( (X, Y) \xrightarrow{(f, g)} (X', Y') \right) \triangleq \mathbf{C}(X, Y) \xrightarrow{\text{Hom}_{\mathbf{C}}(f, g)} \mathbf{C}(X', Y')$$

$\text{Hom}_{\mathbf{C}}(f, g) h \triangleq g \circ h \circ f$

If  $(f, g) : (X, Y) \rightarrow (X', Y')$  in  $\mathbf{C}^{\text{op}} \times \mathbf{C}$  and  $h : X' \rightarrow Y$  in  $\mathbf{C}$ ,  
then in  $\mathbf{C}$  we have  $f : X' \rightarrow X$ ,  $g : Y \rightarrow Y'$  and so  $g \circ h \circ f : X' \rightarrow Y'$



# Natural isomorphisms

Given functors  $F, G : \mathbf{C} \rightarrow \mathbf{D}$ , a **natural isomorphism**  $\theta : F \cong G$  is simply an isomorphism between  $F$  and  $G$  in the functor category  $\mathbf{D}^{\mathbf{C}}$ .

# Natural isomorphisms

Given functors  $F, G : \mathbf{C} \rightarrow \mathbf{D}$ , a **natural isomorphism**  $\theta : F \cong G$  is simply an isomorphism between  $F$  and  $G$  in the functor category  $\mathbf{D}^{\mathbf{C}}$ .

**Lemma.** If  $\theta : F \rightarrow G$  is a natural transformation and for each  $X \in \mathbf{C}$ ,  $\theta_X : FX \rightarrow GX$  is an isomorphism in  $\mathbf{D}$ , then the family of morphisms  $(\theta_X^{-1} : GX \rightarrow FX \mid X \in \mathbf{C})$  gives a natural transformation  $\theta^{-1} : G \rightarrow F$  which is inverse to  $\theta$  in  $\mathbf{D}^{\mathbf{C}}$  and hence  $\theta$  is a natural isomorphism.  $\square$

An adjunction between locally small categories  $\mathbf{C}$  and  $\mathbf{D}$  is simply a triple  $(F, G, \theta)$  where

►  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D}$

►  $\theta$  is a natural isomorphism between the functors

$$\begin{array}{ccccc} & & \mathbf{D}^{\text{op}} \times \mathbf{D} & & \\ & \nearrow^{F^{\text{op}} \times \text{id}_{\mathbf{D}}} & & \searrow^{\text{Hom}_{\mathbf{D}}} & \\ \mathbf{C}^{\text{op}} \times \mathbf{D} & & & & \mathbf{Set} \\ & \searrow_{\text{id}_{\mathbf{C}^{\text{op}}} \times G} & & \nearrow_{\text{Hom}_{\mathbf{C}}} & \\ & & \mathbf{C}^{\text{op}} \times \mathbf{C} & & \end{array}$$

and

## Terminology:

Given  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D}$

is there is some natural isomorphism

$$\theta : \text{Hom}_{\mathbf{D}} \circ (F^{\text{op}} \times \text{id}_{\mathbf{D}}) \cong \text{Hom}_{\mathbf{C}} \circ (\text{id}_{\mathbf{C}^{\text{op}}} \times G)$$

one says

$F$  is a **left adjoint** for  $G$

$G$  is a **right adjoint** for  $F$

and writes

$$F \dashv G$$

**Notation** associated with an adjunction  $(F, G, \theta)$

Given  $\begin{cases} g : FX \rightarrow Y \\ f : X \rightarrow GY \end{cases}$

we write  $\begin{cases} \bar{g} \triangleq \theta_{X,Y}(g) : X \rightarrow GY \\ \bar{f} \triangleq \theta_{X,Y}^{-1}(f) : FX \rightarrow Y \end{cases}$

Thus  $\overline{\bar{g}} = g$ ,  $\overline{\bar{f}} = f$  and naturality of  $\theta_{X,Y}$  in  $X$  and  $Y$  means that

$$\overline{v \circ g \circ Fu} = Gv \circ \bar{g} \circ u$$

**Notation** associated with an adjunction  $(F, G, \theta)$

The existence of  $\theta$  is sometimes indicated by writing

$$\frac{FX \xrightarrow{g} Y}{X \xrightarrow{\bar{g}} GY}$$

Using this notation, one can split the naturality condition for  $\theta$  into two:

$$\frac{FX' \xrightarrow{Fu} FX \xrightarrow{g} Y}{X' \xrightarrow{u} X \xrightarrow{\bar{g}} GY}$$

$$\frac{FX \xrightarrow{g} Y \xrightarrow{v} Y'}{X \xrightarrow{\bar{g}} GY \xrightarrow{Gv} GY'}$$

## Lecture 13

## Recall:

Given categories and functors  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D}$ ,

an adjunction  $F \dashv G$  is specified by functions

$$\theta_{X,Y} \downarrow \frac{FX \xrightarrow{g} Y}{X \xrightarrow{\bar{g}} GY} \qquad \uparrow \theta_{X,Y}^{-1} \frac{FX \xrightarrow{\bar{f}} Y}{X \xrightarrow{f} GY}$$

(for each  $X \in \mathbf{C}$  and  $Y \in \mathbf{D}$ ) satisfying  $\bar{\bar{f}} = f$ ,  $\bar{\bar{g}} = g$  and

$$\frac{FX' \xrightarrow{Fu} FX \xrightarrow{g} Y}{X' \xrightarrow{u} X \xrightarrow{\bar{g}} GY} \qquad \frac{FX \xrightarrow{g} Y \xrightarrow{v} Y'}{X \xrightarrow{\bar{g}} GY \xrightarrow{Gv} GY'}$$



**Theorem.** A category  $\mathbf{C}$  has binary products iff the diagonal functor  $\Delta = \langle \text{id}_{\mathbf{C}}, \text{id}_{\mathbf{C}} \rangle : \mathbf{C} \rightarrow \mathbf{C} \times \mathbf{C}$  has a right adjoint.

**Theorem.** A category  $\mathbf{C}$  with binary products also has all exponentials of pairs of objects iff for all  $X \in \mathbf{C}$ , the functor  $(\_) \times X : \mathbf{C} \rightarrow \mathbf{C}$  has a right adjoint.

**Theorem.** A category  $\mathbf{C}$  has binary products iff the diagonal functor  $\Delta = \langle \text{id}_{\mathbf{C}}, \text{id}_{\mathbf{C}} \rangle : \mathbf{C} \rightarrow \mathbf{C} \times \mathbf{C}$  has a right adjoint.

**Theorem.** A category  $\mathbf{C}$  with binary products also has all exponentials of pairs of objects iff for all  $X \in \mathbf{C}$ , the functor  $(\_)\times X : \mathbf{C} \rightarrow \mathbf{C}$  has a right adjoint.

Both these theorems are instances of the following theorem, a very useful characterisation of when a functor has a right adjoint (or dually, a left adjoint).

# Characterisation of right adjoints

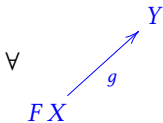
**Theorem.** A functor  $F : \mathbf{C} \rightarrow \mathbf{D}$  has a right adjoint iff for all  $\mathbf{D}$ -objects  $Y \in \mathbf{D}$ , there is a  $\mathbf{C}$ -object  $G Y \in \mathbf{C}$  and a  $\mathbf{D}$ -morphism  $\varepsilon_Y : F(G Y) \rightarrow Y$  with the following “universal property”:

(UP) for all  $X \in \mathbf{C}$  and  $g \in \mathbf{D}(F X, Y)$   
there is a unique  $\bar{g} \in \mathbf{C}(X, G Y)$   
satisfying  $\varepsilon_Y \circ F(\bar{g}) = g$

# Characterisation of right adjoints

**Theorem.** A functor  $F : \mathbf{C} \rightarrow \mathbf{D}$  has a right adjoint iff for all  $\mathbf{D}$ -objects  $Y \in \mathbf{D}$ , there is a  $\mathbf{C}$ -object  $G Y \in \mathbf{C}$  and a  $\mathbf{D}$ -morphism  $\varepsilon_Y : F(G Y) \rightarrow Y$  with the following “universal property”:

(UP) for all  $X \in \mathbf{C}$  and  $g \in \mathbf{D}(F X, Y)$   
there is a unique  $\bar{g} \in \mathbf{C}(X, G Y)$   
satisfying  $\varepsilon_Y \circ F(\bar{g}) = g$



# Characterisation of right adjoints

**Theorem.** A functor  $F : \mathbf{C} \rightarrow \mathbf{D}$  has a right adjoint iff for all  $\mathbf{D}$ -objects  $Y \in \mathbf{D}$ , there is a  $\mathbf{C}$ -object  $GY \in \mathbf{C}$  and a  $\mathbf{D}$ -morphism  $\varepsilon_Y : F(GY) \rightarrow Y$  with the following “universal property”:

(UP) for all  $X \in \mathbf{C}$  and  $g \in \mathbf{D}(FX, Y)$   
there is a unique  $\bar{g} \in \mathbf{C}(X, GY)$   
satisfying  $\varepsilon_Y \circ F(\bar{g}) = g$



# Characterisation of right adjoints

**Theorem.** A functor  $F : \mathbf{C} \rightarrow \mathbf{D}$  has a right adjoint iff for all  $\mathbf{D}$ -objects  $Y \in \mathbf{D}$ , there is a  $\mathbf{C}$ -object  $GY \in \mathbf{C}$  and a  $\mathbf{D}$ -morphism  $\varepsilon_Y : F(GY) \rightarrow Y$  with the following “universal property”:

(UP) for all  $X \in \mathbf{C}$  and  $g \in \mathbf{D}(FX, Y)$   
there is a unique  $\bar{g} \in \mathbf{C}(X, GY)$   
satisfying  $\varepsilon_Y \circ F(\bar{g}) = g$

$$\forall \begin{array}{ccc} & & Y \\ & \nearrow g & \\ FX & & \end{array}$$

$$\exists! \begin{array}{ccc} & GY & \\ & \uparrow \bar{g} & \\ & X & \end{array}$$

$$\text{with } \begin{array}{ccc} & F(GY) & \xrightarrow{\varepsilon_Y} & Y \\ & \uparrow F\bar{g} & \nearrow g & \\ & FX & & \end{array}$$

## Proof of the Theorem—“only if” part:

Given an adjunction  $(F, G, \theta)$ , for each  $Y \in \mathbf{D}$  we produce  $\varepsilon_Y : F(GY) \rightarrow Y$  in  $\mathbf{D}$  satisfying (UP).

## Proof of the Theorem—“only if” part:

Given an adjunction  $(F, G, \theta)$ , for each  $Y \in \mathbf{D}$  we produce  $\varepsilon_Y : F(GY) \rightarrow Y$  in  $\mathbf{D}$  satisfying (UP).

We are given  $\theta_{X,Y} : \mathbf{D}(FX, Y) \cong \mathbf{C}(X, GY)$ , natural in  $X$  and  $Y$ . Define

$$\varepsilon_Y \triangleq \theta_{GY, Y}^{-1}(\text{id}_{GY}) : F(GY) \rightarrow Y$$

In other words  $\varepsilon_Y = \overline{\text{id}_{GY}}$ .



## Proof of the Theorem—“only if” part:

Given an adjunction  $(F, G, \theta)$ , for each  $Y \in \mathbf{D}$  we produce  $\varepsilon_Y : F(GY) \rightarrow Y$  in  $\mathbf{D}$  satisfying (UP).

We are given  $\theta_{X,Y} : \mathbf{D}(FX, Y) \cong \mathbf{C}(X, GY)$ , natural in  $X$  and  $Y$ . Define

$$\varepsilon_Y \triangleq \theta_{GY, Y}^{-1}(\text{id}_{GY}) : F(GY) \rightarrow Y$$

In other words  $\varepsilon_Y = \overline{\text{id}_{GY}}$ .

Given any  $\begin{cases} g : FX \rightarrow Y & \text{in } \mathbf{D} \\ f : X \rightarrow GY & \text{in } \mathbf{C} \end{cases}$ , by naturality of  $\theta$  we have

$$\frac{FX \xrightarrow{g} Y}{X \xrightarrow{\bar{g}} GY} \text{ and } \frac{\varepsilon_Y \circ Ff : FX \xrightarrow{Ff} F(GY) \xrightarrow{\overline{\text{id}_{GY}}} Y}{f : X \xrightarrow{f} GY \xrightarrow{\text{id}_{GY}} GY}$$

Hence  $g = \varepsilon_Y \circ F\bar{g}$  and  $g = \varepsilon_Y \circ Ff \Rightarrow \bar{g} = f$ .

Thus we do indeed have (UP).

## Proof of the Theorem—“if” part:

We are given  $F : \mathbf{C} \rightarrow \mathbf{D}$  and for each  $Y \in \mathbf{D}$  a  $\mathbf{C}$ -object  $GY$  and  $\mathbf{C}$ -morphism  $\varepsilon_Y : F(GY) \rightarrow Y$  satisfying (UP). We have to

1. extend  $Y \mapsto GY$  to a functor  $G : \mathbf{D} \rightarrow \mathbf{C}$
2. construct a natural isomorphism  $\theta : \text{Hom}_{\mathbf{D}} \circ (F^{\text{op}} \times \text{id}_{\mathbf{D}}) \cong \text{Hom}_{\mathbf{C}} \circ (\text{id}_{\mathbf{C}^{\text{op}}} \times G)$

## Proof of the Theorem—“if” part:

We are given  $F : \mathbf{C} \rightarrow \mathbf{D}$  and for each  $Y \in \mathbf{D}$  a  $\mathbf{C}$ -object  $GY$  and  $\mathbf{C}$ -morphism  $\varepsilon_Y : F(GY) \rightarrow Y$  satisfying (UP). We have to

1. extend  $Y \mapsto GY$  to a functor  $G : \mathbf{D} \rightarrow \mathbf{C}$

For each  $\mathbf{D}$ -morphism  $g : Y' \rightarrow Y$  we get  $F(GY') \xrightarrow{\varepsilon_{Y'}} Y' \xrightarrow{g} Y$  and can apply (UP) to get

$$Gg \triangleq \overline{g \circ \varepsilon_{Y'}} : GY' \rightarrow GY$$

The uniqueness part of (UP) implies

$$G \text{id} = \text{id} \quad \text{and} \quad G(g' \circ g) = Gg' \circ Gg$$

so that we get a functor  $G : \mathbf{D} \rightarrow \mathbf{C}$ .  $\square$

## Proof of the Theorem—“if” part:

We are given  $F : \mathbf{C} \rightarrow \mathbf{D}$  and for each  $Y \in \mathbf{D}$  a  $\mathbf{C}$ -object  $GY$  and  $\mathbf{C}$ -morphism  $\varepsilon_Y : F(GY) \rightarrow Y$  satisfying (UP). We have to

2. *construct a natural isomorphism  $\theta : \text{Hom}_{\mathbf{D}} \circ (F^{\text{op}} \times \text{id}_{\mathbf{D}}) \cong \text{Hom}_{\mathbf{C}} \circ (\text{id}_{\mathbf{C}^{\text{op}}} \times G)$*

Since for all  $g : FX \rightarrow Y$  there is a unique  $f : X \rightarrow GY$  with  $g = \varepsilon_Y \circ Ff$ ,

$$f \mapsto \bar{f} \triangleq \varepsilon_Y \circ Ff$$

determines a bijection  $\mathbf{C}(X, GY) \cong \mathbf{C}(FX, Y)$ ; and it is natural in  $X$  &  $Y$  because

$$\begin{aligned} \overline{Gv \circ f \circ u} &\triangleq \varepsilon_{Y'} \circ F(Gv \circ f \circ u) \\ &= (\varepsilon_{Y'} \circ F(Gv)) \circ Ff \circ Fu && \text{since } F \text{ is a functor} \\ &= (v \circ \varepsilon_Y) \circ Ff \circ Fu && \text{by definition of } Gv \\ &= v \circ \bar{f} \circ Fu && \text{by definition of } \bar{f} \end{aligned}$$

So we can take  $\theta$  to be the inverse of this natural isomorphism.  $\square$

## Dual of the Theorem:

$G : \mathbf{C} \leftarrow \mathbf{D}$  has a **left** adjoint iff for all  $X \in \mathbf{C}$  there are  $FX \in \mathbf{D}$  and  $\eta_X \in \mathbf{C}(X, G(FX))$  with the universal property:

for all  $Y \in \mathbf{D}$  and  $f \in \mathbf{C}(X, GY)$   
there is a unique  $\bar{f} \in \mathbf{D}(FX, Y)$   
satisfying  $G\bar{f} \circ \eta_X = f$

## Dual of the Theorem:

$G : \mathbf{C} \leftarrow \mathbf{D}$  has a **left** adjoint iff for all  $X \in \mathbf{C}$  there are  $F X \in \mathbf{D}$  and  $\eta_X \in \mathbf{C}(X, G(F X))$  with the universal property:

for all  $Y \in \mathbf{D}$  and  $f \in \mathbf{C}(X, G Y)$   
there is a unique  $\bar{f} \in \mathbf{D}(F X, Y)$   
satisfying  $G \bar{f} \circ \eta_X = f$

E.g. we can conclude that **the forgetful functor**  $U : \mathbf{Mon} \rightarrow \mathbf{Set}$  has a **left adjoint**  $F : \mathbf{Set} \rightarrow \mathbf{Mon}$ , because of the universal property of

$$F \Sigma \triangleq (\text{List } \Sigma, @, \text{nil}) \quad \text{and} \quad \eta_\Sigma : \Sigma \rightarrow \text{List } \Sigma$$

noted in Lecture 3.

# Why are adjoint functors important/useful?

Their universal property (UP) usually embodies some useful mathematical construction

(e.g. “freely generated structures are left adjoints for forgetting-structure”)  
and pins it down uniquely up to isomorphism.

# Lecture 14



# Dependent Types

A brief look at some category theory for modelling type theories with **dependent types**.

Will restrict attention to the case of **Set**, rather than in full generality.

Further reading:

M. Hofmann, *Syntax and Semantics of Dependent Types*. In: A.M. Pitts and P. Dybjer (eds), *Semantics and Logics of Computation* (CUP, 1997).

## Simple types

$$\diamond, x_1 : T_1, \dots, x_n : T_n \vdash t(x_1, \dots, x_n) : T$$

## Dependent types

$$\diamond, x_1 : T_1, \dots, x_n : T_n \vdash t(x_1, \dots, x_n) : T(x_1, \dots, x_n)$$

and more generally

$$\diamond, x_1 : T_1, x_2 : T_2(x_1), x_3 : T_3(x_1, x_2), \dots \vdash \\ t(x_1, x_2, x_3, \dots) : T(x_1, x_2, x_3, \dots)$$

If type expressions denote sets, then

a type  $T_1(x)$  dependent upon  $x : T$

should denote

an indexed family of sets  $(E_i \mid i \in I)$   
(where  $I$  is the set denoted by type  $T$ )

i.e.  $E : I \rightarrow \mathbf{Set}$  is a set-valued function on a set  $I$ .

For each  $I \in \mathbf{Set}$ , let  $\mathbf{Set}^I$  be the category with

- ▶  $\mathbf{obj}(\mathbf{Set}^I) \triangleq (\mathbf{obj} \mathbf{Set})^I$ , so objects are  $I$ -indexed families of sets,  $X = (X_i \mid i \in I)$
- ▶ morphisms  $f : X \rightarrow Y$  in  $\mathbf{Set}^I$  are  $I$ -indexed families of functions  $f = (f_i \in \mathbf{Set}(X_i, Y_i) \mid i \in I)$
- ▶ composition:  $(g \circ f) \triangleq (g_i \circ f_i \mid i \in I)$   
(i.e. use composition of functions in  $\mathbf{Set}$  at each index  $i \in I$ )
- ▶ identity:  $\mathbf{id}_X \triangleq (\mathbf{id}_{X_i} \mid i \in I)$   
(i.e. use identity functions in  $\mathbf{Set}$  at each index  $i \in I$ )

For each  $p : I \rightarrow J$  in **Set**, let  $p^* : \mathbf{Set}^J \rightarrow \mathbf{Set}^I$  be the functor defined by:

$$p^* \left( \begin{array}{c} Y_j \\ \downarrow f_j \\ Y'_j \end{array} \middle| j \in J \right) \triangleq \left( \begin{array}{c} Y_{p i} \\ \downarrow f_{p i} \\ Y'_{p i} \end{array} \middle| i \in I \right)$$

i.e.  $p^*$  takes  $J$ -indexed families of sets/functions to  $I$ -indexed ones by precomposing with  $p$

# Dependent products

## of families of sets

For  $I, J \in \mathbf{Set}$ , consider the functor  $\pi_1^* : \mathbf{Set}^I \rightarrow \mathbf{Set}^{I \times J}$  induced by precomposition with the first projection function  $\pi_1 : I \times J \rightarrow I$ .

**Theorem.**  $\pi_1^*$  has a left adjoint  $\Sigma : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

**Proof.** We apply the Theorem from Lecture 13: for each  $E \in \mathbf{Set}^{I \times J}$  we define  $\Sigma E \in \mathbf{Set}^I$  and  $\eta_E : E \rightarrow \pi_1^*(\Sigma E)$  in  $\mathbf{Set}^{I \times J}$  with the required universal property...

**Theorem.**  $\pi_1^*$  has a left adjoint  $\Sigma : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

For each  $E \in \mathbf{Set}^{I \times J}$ , define  $\Sigma E \in \mathbf{Set}^I$  to be the function mapping each  $i \in I$  to the set

$$(\Sigma E)_i \triangleq \sum_{j \in J} E_{(i,j)} = \{(j, e) \mid j \in J \wedge e \in E_{(i,j)}\}$$

**Theorem.**  $\pi_1^*$  has a left adjoint  $\Sigma : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

For each  $E \in \mathbf{Set}^{I \times J}$ , define  $\Sigma E \in \mathbf{Set}^I$  to be the function mapping each  $i \in I$  to the set

$$(\Sigma E)_i \triangleq \sum_{j \in J} E_{(i,j)} = \{(j, e) \mid j \in J \wedge e \in E_{(i,j)}\}$$

and define  $\eta_E : E \rightarrow \pi_1^*(\Sigma E)$  in  $\mathbf{Set}^{I \times J}$  to be the function mapping each  $(i, j) \in I \times J$  to the function  $(\eta_E)_{(i,j)} : E_{(i,j)} \rightarrow (\Sigma E)_i$  given by  $e \mapsto (j, e)$ .

**Universal property–**



**Theorem.**  $\pi_1^*$  has a left adjoint  $\Sigma : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

For each  $E \in \mathbf{Set}^{I \times J}$ , define  $\Sigma E \in \mathbf{Set}^I$  to be the function mapping each  $i \in I$  to the set

$$(\Sigma E)_i \triangleq \sum_{j \in J} E_{(i,j)} = \{(j, e) \mid j \in J \wedge e \in E_{(i,j)}\}$$

and define  $\eta_E : E \rightarrow \pi_1^*(\Sigma E)$  in  $\mathbf{Set}^{I \times J}$  to be the function mapping each  $(i, j) \in I \times J$  to the function  $(\eta_E)_{(i,j)} : E_{(i,j)} \rightarrow (\Sigma E)_i$  given by  $e \mapsto (j, e)$ .

**Universal property**–*existence part*: given any  $X \in \mathbf{Set}^I$  and  $f : E \rightarrow \pi_1^*(X)$  in  $\mathbf{Set}^{I \times J}$ , we have

$$\begin{array}{ccc}
 E & \xrightarrow{\eta_E} & \pi_1^*(\Sigma E) & & \Sigma E \\
 & \searrow f & \downarrow \pi_1^*(\bar{f}) & & \downarrow \bar{f} \\
 & & \pi_1^*(X) & & X
 \end{array}$$

where for all  $i \in I, j \in J$  and  $e \in E_{(i,j)}$   $\bar{f}_i(j, e) \triangleq f_{(i,j)}(e)$

**Theorem.**  $\pi_1^*$  has a left adjoint  $\Sigma : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

For each  $E \in \mathbf{Set}^{I \times J}$ , define  $\Sigma E \in \mathbf{Set}^I$  to be the function mapping each  $i \in I$  to the set

$$(\Sigma E)_i \triangleq \sum_{j \in J} E_{(i,j)} = \{(j, e) \mid j \in J \wedge e \in E_{(i,j)}\}$$

and define  $\eta_E : E \rightarrow \pi_1^*(\Sigma E)$  in  $\mathbf{Set}^{I \times J}$  to be the function mapping each  $(i, j) \in I \times J$  to the function  $(\eta_E)_{(i,j)} : E_{(i,j)} \rightarrow (\Sigma E)_i$  given by  $e \mapsto (j, e)$ .

**Universal property–uniqueness part:** given  $g : \Sigma E \rightarrow X$  in  $\mathbf{Set}^I$  making

$$\begin{array}{ccc} E & \xrightarrow{\eta_E} & \pi_1^*(\Sigma E) \\ & \searrow f & \downarrow \pi_1^*(g) \\ & & \pi_1^*(X) \end{array} \quad \text{commute in } \mathbf{Set}^{I \times J},$$

then for all  $i \in I$ , and  $(j, e) \in (\Sigma E)_i$  we have

$$\overline{f}_i(j, e) \triangleq f_{(i,j)}(e) = (\pi_1^*g \circ \eta_E)_{(i,j)} e = (\pi_1^*g)_{(i,j)}((\eta_E)_{(i,j)} e) \triangleq g_i(j, e)$$

so  $g = \overline{f}$ .  $\square$

# Dependent functions

## of families of sets

We have seen that the left adjoint to  $\pi_1^* : \mathbf{Set}^I \rightarrow \mathbf{Set}^{I \times J}$  is given by dependent products of sets.

Dually, dependent function sets give:

**Theorem.**  $\pi_1^*$  has a right adjoint  $\Pi : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

**Proof.** We apply the Theorem from Lecture 13: for each  $E \in \mathbf{Set}^{I \times J}$  we define  $\Pi E \in \mathbf{Set}^I$  and  $\varepsilon_E : \pi_1^*(\Pi E) \rightarrow E$  in  $\mathbf{Set}^{I \times J}$  with the required universal property...

**Theorem.**  $\pi_1^*$  has a right adjoint  $\Pi : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

For each  $E \in \mathbf{Set}^{I \times J}$ , define  $\Pi E \in \mathbf{Set}^I$  to be the function mapping each  $i \in I$  to the set

$$(\Pi E)_i \triangleq \prod_{j \in J} E_{(i,j)} = \{f \subseteq (\Sigma E)_i \mid f \text{ is single-value and total}\}$$

where  $f \subseteq (\Sigma E)_i$  is

**single-valued** if  $\forall j \in J, \forall e, e' \in E_{(i,j)}, (j, e) \in f \wedge (j, e') \in f \Rightarrow e = e'$

**total** if  $\forall j \in J, \exists e \in E_{(i,j)} (j, e) \in f$

Thus each  $f \in (\Pi E)_i$  is a **dependently typed function** mapping elements  $j \in J$  to elements of  $E_{(i,j)}$  (result set depends on the argument  $j$ ).

**Theorem.**  $\pi_1^*$  has a right adjoint  $\Pi : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

For each  $E \in \mathbf{Set}^{I \times J}$ , define  $\Pi E \in \mathbf{Set}^I$  to be the function mapping each  $i \in I$  to the set

$$(\Pi E)_i \triangleq \prod_{j \in J} E_{(i,j)} = \{f \subseteq (\Sigma E)_i \mid f \text{ is single-value and total}\}$$

and define  $\varepsilon_E : \pi_1^*(\Pi E) \rightarrow E$  in  $\mathbf{Set}^{I \times J}$  to be the function mapping each  $(i, j) \in I \times J$  to the function  $(\varepsilon_E)_{(i,j)} : (\Pi E)_i \rightarrow E_{(i,j)}$  given by  $f \mapsto f j = \text{unique } e \in E_{(i,j)} \text{ such that } (j, e) \in f$ .

**Universal property–**

**Theorem.**  $\pi_1^*$  has a right adjoint  $\Pi : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

For each  $E \in \mathbf{Set}^{I \times J}$ , define  $\Pi E \in \mathbf{Set}^I$  to be the function mapping each  $i \in I$  to the set

$$(\Pi E)_i \triangleq \prod_{j \in J} E_{(i,j)} = \{f \subseteq (\Sigma E)_i \mid f \text{ is single-value and total}\}$$

and define  $\varepsilon_E : \pi_1^*(\Pi E) \rightarrow E$  in  $\mathbf{Set}^{I \times J}$  to be the function mapping each  $(i, j) \in I \times J$  to the function  $(\varepsilon_E)_{(i,j)} : (\Pi E)_i \rightarrow E_{(i,j)}$  given by  $f \mapsto f j = \text{unique } e \in E_{(i,j)} \text{ such that } (j, e) \in f$ .

**Universal property–existence part:** given any  $X \in \mathbf{Set}^I$  and  $f : \pi_1^*(X) \rightarrow E$  in  $\mathbf{Set}^{I \times J}$ , we have

$$\begin{array}{ccc}
 \Pi E & & \pi_1^*(\Pi E) \xrightarrow{\varepsilon_E} E \\
 \uparrow \bar{f} & & \uparrow \\
 X & & \pi_1^*(X) \xrightarrow{f} E
 \end{array}$$

where for all  $i \in I$  and  $x \in X_i$   $\bar{f}_i x \triangleq \{(j, f_{(i,j)} x) \mid j \in J\}$

**Theorem.**  $\pi_1^*$  has a right adjoint  $\Pi : \mathbf{Set}^{I \times J} \rightarrow \mathbf{Set}^I$ .

For each  $E \in \mathbf{Set}^{I \times J}$ , define  $\Pi E \in \mathbf{Set}^I$  to be the function mapping each  $i \in I$  to the set

$$(\Pi E)_i \triangleq \prod_{j \in J} E_{(i,j)} = \{f \subseteq (\Sigma E)_i \mid f \text{ is single-value and total}\}$$

and define  $\varepsilon_E : \pi_1^*(\Pi E) \rightarrow E$  in  $\mathbf{Set}^{I \times J}$  to be the function mapping each  $(i, j) \in I \times J$  to the function  $(\varepsilon_E)_{(i,j)} : (\Pi E)_i \rightarrow E_{(i,j)}$  given by  $f \mapsto f j = \text{unique } e \in E_{(i,j)} \text{ such that } (j, e) \in f$ .

**Universal property—uniqueness part:** given  $g : X \rightarrow \Pi E$  in  $\mathbf{Set}^I$  making

$$\begin{array}{ccc} \pi_1^*(\Pi E) & \xrightarrow{\varepsilon_E} & E \\ \pi_1^*(g) \uparrow & \nearrow f & \\ \pi_1^*(X) & & \end{array} \text{ commute in } \mathbf{Set}^{I \times J},$$

then for all  $i \in I, j \in J$  and  $x \in X_i$  we have

$$\overline{f}_i x j \triangleq f_{(i,j)} x = (\varepsilon_E \circ \pi_1^* g)_{(i,j)} x = (\varepsilon_E)_{(i,j)} (g_i x) \triangleq g_i x j$$

so  $g = \overline{f}$ .  $\square$

# Isomorphism of categories

Two categories **C** and **D** are **isomorphic** if they are isomorphic objects in the category of all categories of some given size, that is, if there are functors

$$\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D} \text{ with } \text{id}_{\mathbf{C}} = G \circ F \text{ and } F \circ G = \text{id}_{\mathbf{D}}.$$

In which case, as usual, we write  $\mathbf{C} \cong \mathbf{D}$ .



# Equivalence of categories

Two categories  $\mathbf{C}$  and  $\mathbf{D}$  are **equivalent** if there are

functors  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D}$  and natural isomorphisms

$\eta : \text{id}_{\mathbf{C}} \cong G \circ F$  and  $\varepsilon : F \circ G \cong \text{id}_{\mathbf{D}}$ .

In which case, one writes  $\mathbf{C} \simeq \mathbf{D}$ .

# Equivalence of categories

Two categories  $\mathbf{C}$  and  $\mathbf{D}$  are **equivalent** if there are

functors  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D}$  and natural isomorphisms

$\eta : \text{id}_{\mathbf{C}} \cong G \circ F$  and  $\varepsilon : F \circ G \cong \text{id}_{\mathbf{D}}$ .

In which case, one writes  $\mathbf{C} \simeq \mathbf{D}$ .

Some deep results in mathematics take the form of equivalences of categories.

E.g.

Stone duality:  $\left( \begin{array}{c} \text{category of} \\ \text{Boolean algebras} \end{array} \right)^{\text{op}} \simeq \left( \begin{array}{c} \text{category of compact} \\ \text{totally disconnected} \\ \text{Hausdorff spaces} \end{array} \right)$

Gelfand duality:  $\left( \begin{array}{c} \text{category of} \\ \text{abelian } C^* \text{ algebras} \end{array} \right)^{\text{op}} \simeq \left( \begin{array}{c} \text{category of compact} \\ \text{Hausdorff spaces} \end{array} \right)$

# Example: $\mathbf{Set}^I \simeq \mathbf{Set}/I$

$\mathbf{Set}/I$  is a **slice category**:

- ▶ objects are pairs  $(E, p)$  where  $E \in \text{obj } \mathbf{Set}$  and  $p \in \mathbf{Set}(E, I)$
- ▶ morphisms  $g : (E, p) \rightarrow (E', p')$  are  $f \in \mathbf{Set}(E, E')$  satisfying  $p' \circ f = p$  in  $\mathbf{Set}$
- ▶ composition and identities – as for  $\mathbf{Set}$

## Example: $\mathbf{Set}^I \simeq \mathbf{Set}/I$

There are functors  $F : \mathbf{Set}^I \rightarrow \mathbf{Set}/I$  and  $G : \mathbf{Set}/I \rightarrow \mathbf{Set}^I$ , given on objects and morphisms by:

$$FX \triangleq (\{(i, x) \mid i \in I \wedge x \in X_i\}, \text{fst})$$

$$Ff(i, x) \triangleq (i, f_i x)$$

$$G(E, p) \triangleq (\{e \in E \mid p e = i\} \mid i \in I)$$

$$(Gf)_i e \triangleq f e$$

## Example: $\mathbf{Set}^I \simeq \mathbf{Set}/I$

There are functors  $F : \mathbf{Set}^I \rightarrow \mathbf{Set}/I$  and  $G : \mathbf{Set}/I \rightarrow \mathbf{Set}^I$ , given on objects and morphisms by:

$$\begin{aligned}FX &\triangleq (\{(i, x) \mid i \in I \wedge x \in X_i\}, \text{fst}) \\ Ff(i, x) &\triangleq (i, f_i x) \\ G(E, p) &\triangleq (\{e \in E \mid p e = i\} \mid i \in I) \\ (Gf)_i e &\triangleq f e\end{aligned}$$

There are natural isomorphisms

$$\eta : \text{id}_{\mathbf{Set}^I} \cong G \circ F \text{ and } \varepsilon : F \circ G \cong \text{id}_{\mathbf{Set}/I}$$

defined by... [exercise]

**FACT** Given  $p : I \rightarrow J$  in **Set**, the composition

$$\mathbf{Set}/J \simeq \mathbf{Set}^J \xrightarrow{p^*} \mathbf{Set}^I \simeq \mathbf{Set}/I$$

is the functor “**pullback** along  $p$ ”.

One can generalize from **Set** to any category **C** with pullbacks and model  $\Sigma/\Pi$  types by left/right adjoints to pullback functors – see **locally cartesian closed** categories in the literature.

# Lecture 15

# Presheaf categories

Let  $\mathbf{C}$  be a small category. The functor category  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is called the category of presheaves on  $\mathbf{C}$ .

- ▶ objects are contravariant functors from  $\mathbf{C}$  to  $\mathbf{Set}$
- ▶ morphisms are natural transformations

Much used in the semantics of various dependently-typed languages and logics.



# Yoneda functor

$$\mathbf{y} : \mathbf{C} \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}$$

(where  $\mathbf{C}$  is a small category)

is the Curried version of the hom functor

$$\mathbf{C} \times \mathbf{C}^{\text{op}} \cong \mathbf{C}^{\text{op}} \times \mathbf{C} \xrightarrow{\text{Hom}_{\mathbf{C}}} \mathbf{Set}$$

# Yoneda functor

$$\mathcal{Y} : \mathbf{C} \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}$$

(where  $\mathbf{C}$  is a small category)

is the Curried version of the hom functor

$$\mathbf{C} \times \mathbf{C}^{\text{op}} \cong \mathbf{C}^{\text{op}} \times \mathbf{C} \xrightarrow{\text{Hom}_{\mathbf{C}}} \mathbf{Set}$$

- ▶ For each  $\mathbf{C}$ -object  $X$ , the object  $\mathcal{Y}X \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is the functor  $\mathbf{C}(\_, X) : \mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$  given by

$$\begin{array}{ccccc} Z & \mapsto & \mathbf{C}(Z, X) & & g \circ f \\ \downarrow f & \mapsto & \uparrow & & \uparrow \\ Y & \mapsto & \mathbf{C}(Y, X) & & g \end{array}$$

# Yoneda functor

$$\mathcal{Y} : \mathbf{C} \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}$$

(where  $\mathbf{C}$  is a small category)

is the Curried version of the hom functor

$$\mathbf{C} \times \mathbf{C}^{\text{op}} \cong \mathbf{C}^{\text{op}} \times \mathbf{C} \xrightarrow{\text{Hom}_{\mathbf{C}}} \mathbf{Set}$$

- ▶ For each  $\mathbf{C}$ -object  $X$ , the object  $\mathcal{Y}X \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is the functor  $\mathbf{C}(\_, X) : \mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$  given by

$$\begin{array}{ccc} Z & \mapsto & \mathbf{C}(Z, X) \\ \downarrow f & \mapsto & \uparrow \\ Y & \mapsto & \mathbf{C}(Y, X) \end{array} \quad \begin{array}{c} g \circ f \\ \uparrow f^* \\ g \end{array}$$

this function is often written as  $f^*$

# Yoneda functor

$$\mathbf{y} : \mathbf{C} \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}$$

(where  $\mathbf{C}$  is a small category)

is the Curried version of the hom functor

$$\mathbf{C} \times \mathbf{C}^{\text{op}} \cong \mathbf{C}^{\text{op}} \times \mathbf{C} \xrightarrow{\text{Hom}_{\mathbf{C}}} \mathbf{Set}$$

- For each  $\mathbf{C}$ -morphism  $Y \xrightarrow{f} X$ , the morphism  $\mathbf{y}Y \xrightarrow{\mathbf{y}f} \mathbf{y}X$  in  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is the natural transformation whose component at any given  $Z \in \mathbf{C}^{\text{op}}$  is the function

$$\begin{array}{ccc} \mathbf{y}Y(Z) & \xrightarrow{(\mathbf{y}f)_Z} & \mathbf{y}X(Z) \\ \parallel & & \parallel \\ \mathbf{C}(Z, Y) & & \mathbf{C}(Z, X) \\ & g \longmapsto & f \circ g \end{array}$$

# Yoneda functor

$$\mathbf{y} : \mathbf{C} \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}$$

(where  $\mathbf{C}$  is a small category)

is the Curried version of the hom functor

$$\mathbf{C} \times \mathbf{C}^{\text{op}} \cong \mathbf{C}^{\text{op}} \times \mathbf{C} \xrightarrow{\text{Hom}_{\mathbf{C}}} \mathbf{Set}$$

- ▶ For each  $\mathbf{C}$ -morphism  $Y \xrightarrow{f} X$ , the morphism  $\mathbf{y}Y \xrightarrow{\mathbf{y}f} \mathbf{y}X$  in  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is the natural transformation whose component at any given  $Z \in \mathbf{C}^{\text{op}}$  is the function

$$\begin{array}{ccc} \mathbf{y}Y(Z) & \xrightarrow{(\mathbf{y}f)_Z} & \mathbf{y}X(Z) \\ \parallel & & \parallel \\ \mathbf{C}(Z, Y) & & \mathbf{C}(Z, X) \end{array}$$

this function is often written as  $f_*$

$$g \mapsto f_* \rightarrow f \circ g$$

# The Yoneda Lemma

For each small category  $\mathbf{C}$ , each object  $X \in \mathbf{C}$  and each presheaf  $F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , there is a bijection of sets

$$\eta_{X,F} : \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) \cong F(X)$$

which is natural in both  $X$  and  $F$ .

# The Yoneda Lemma

For each small category  $\mathbf{C}$ , each object  $X \in \mathbf{C}$  and each presheaf  $F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , there is a bijection of sets

$$\eta_{X,F} : \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) \cong F(X)$$

which is natural in both  $X$  and  $F$ .

the set of natural transformations from  
the functor  $\mathcal{Y}X : \mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$   
to the functor  $F : \mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$

the value of  
 $F : \mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$   
at  $X$

# The Yoneda Lemma

For each small category  $\mathbf{C}$ , each object  $X \in \mathbf{C}$  and each presheaf  $F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , there is a bijection of sets

$$\eta_{X,F} : \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) \cong F(X)$$

which is natural in both  $X$  and  $F$ .

**Definition of the function**  $\eta_{X,F} : \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) \rightarrow F(X)$ :

for each  $\theta : \mathcal{Y}X \rightarrow F$  in  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  we have the function

$\mathbf{C}(X, X) = \mathcal{Y}X(X) \xrightarrow{\theta_X} F(X)$  and define

$$\eta_{X,F}(\theta) \triangleq \theta_X(\text{id}_X)$$



# The Yoneda Lemma

For each small category  $\mathbf{C}$ , each object  $X \in \mathbf{C}$  and each presheaf  $F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , there is a bijection of sets

$$\eta_{X,F} : \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) \cong F(X)$$

which is natural in both  $X$  and  $F$ .

**Definition of the function**  $\eta_{X,F}^{-1} : F(X) \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F)$ :

for each  $x \in F(X)$ ,  $Y \in \mathbf{C}$  and  $f \in \mathcal{Y}X(Y) = \mathbf{C}(Y, X)$ ,

we get a  $F(X) \xrightarrow{F(f)} F(Y)$  in  $\mathbf{Set}$  and hence  $F(f)(x) \in F(Y)$ ;

# The Yoneda Lemma

For each small category  $\mathbf{C}$ , each object  $X \in \mathbf{C}$  and each presheaf  $F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , there is a bijection of sets

$$\eta_{X,F} : \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) \cong F(X)$$

which is natural in both  $X$  and  $F$ .

**Definition of the function**  $\eta_{X,F}^{-1} : F(X) \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F)$ :

for each  $x \in F(X)$ ,  $Y \in \mathbf{C}$  and  $f \in \mathcal{Y}X(Y) = \mathbf{C}(Y, X)$ ,

we get a  $F(X) \xrightarrow{F(f)} F(Y)$  in  $\mathbf{Set}$  and hence  $F(f)(x) \in F(Y)$ ;

Define  $\left(\eta_{X,F}^{-1}(x)\right)_Y : \mathcal{Y}X(Y) \rightarrow F(Y)$  by

$$\left(\eta_{X,F}^{-1}(x)\right)_Y(f) \triangleq F(f)(x)$$

**check** this gives a natural transformation  $\eta_{X,F}^{-1}(x) : \mathcal{Y}X \rightarrow F$

**Proof of**  $\eta_{X,F} \circ \eta_{X,F}^{-1} = \text{id}_{F(X)}$

For any  $x \in F(X)$  we have

$$\begin{aligned}\eta_{X,F} \left( \eta_{X,F}^{-1}(x) \right) &\triangleq \left( \eta_{X,F}^{-1}(x) \right)_X (\text{id}_X) \\ &\triangleq F(\text{id}_X)(x) \\ &= \text{id}_{F(X)}(x) \\ &= x\end{aligned}$$

by definition of  $\eta_{X,F}$

by definition of  $\eta_{X,F}^{-1}$

since  $F$  is a functor

# Proof of $\eta_{X,F}^{-1} \circ \eta_{X,F} = \text{id}_{\text{Set}^{\text{C}^{\text{op}}}(Y, X, F)}$

For any  $Y \xrightarrow{\theta} X$  in  $\text{Set}^{\text{C}^{\text{op}}}$  and  $Z \xrightarrow{f} X$  in  $\mathbf{C}$ , we have

$$\begin{aligned} \left( \eta_{X,F}^{-1}(\eta_{X,F}(\theta)) \right)_Y f &\triangleq \left( \eta_{X,F}^{-1}(\theta_X(\text{id}_X)) \right)_Y f \\ &\triangleq F(f)(\theta_X(\text{id}_X)) \\ &= \theta_Y(f^*(\text{id}_X)) \\ &\triangleq \theta_Y(\text{id}_X \circ f) \\ &= \theta_Y(f) \end{aligned}$$

by definition of  $\eta_{X,F}$

by definition of  $\eta_{X,F}^{-1}$

by naturality of  $\theta$

by definition of  $f^*$

naturality of  $\theta$

$$\begin{array}{ccc} Y & \xrightarrow{\theta_Y} & X \\ \uparrow f^* & & \uparrow F(f) \\ Y & \xrightarrow{\theta_Y} & X \end{array}$$

## Proof of $\eta_{X,F}^{-1} \circ \eta_{X,F} = \text{id}_{\text{Set}^{\text{C}^{\text{op}}}(Y, X, F)}$

For any  $Y \xrightarrow{\theta} X$  in  $\text{Set}^{\text{C}^{\text{op}}}$  and  $Y \xrightarrow{f} X$  in  $\mathbf{C}$ , we have

$$\begin{aligned} \left( \eta_{X,F}^{-1} (\eta_{X,F}(\theta)) \right)_Y f &\triangleq \left( \eta_{X,F}^{-1} (\theta_X(\text{id}_X)) \right)_Y f \\ &\triangleq F(f)(\theta_X(\text{id}_X)) \\ &= \theta_Y(f^*(\text{id}_X)) \\ &\triangleq \theta_Y(\text{id}_X \circ f) \\ &= \theta_Y(f) \end{aligned}$$

by definition of  $\eta_{X,F}$

by definition of  $\eta_{X,F}^{-1}$

by naturality of  $\theta$

by definition of  $f^*$

so  $\forall \theta, Y, \left( \eta_{X,F}^{-1} (\eta_{X,F}(\theta)) \right)_Y = \theta_Y$

so  $\forall \theta, \eta_{X,F}^{-1} (\eta_{X,F}(\theta)) = \theta$

so  $\eta_{X,F}^{-1} \circ \eta_{X,F} = \text{id}$ .

# The Yoneda Lemma

For each small category  $\mathbf{C}$ , each object  $X \in \mathbf{C}$  and each presheaf  $F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , there is a bijection of sets

$$\eta_{X,F} : \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) \cong F(X)$$

which is natural in both  $X$  and  $F$ .

## Proof that $\eta_{X,F}$ is natural in $F$ :

Given  $F \xrightarrow{\varphi} G$  in  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , have to show that

$$\begin{array}{ccc} \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) & \xrightarrow{\eta_{X,F}} & F(X) \\ \varphi_* \downarrow & & \downarrow \varphi_X \\ \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, G) & \xrightarrow{\eta_{X,G}} & G(X) \end{array}$$

commutes in  $\mathbf{Set}$ . For all  $\mathcal{Y}X \xrightarrow{\theta} F$  we have

$$\begin{aligned} \varphi_X(\eta_{X,F}(\theta)) &\triangleq \varphi_X(\theta_X(\text{id}_X)) \\ &\triangleq (\varphi \circ \theta)_X(\text{id}_X) \\ &\triangleq \eta_{X,G}(\varphi \circ \theta) \\ &\triangleq \eta_{X,G}(\varphi_*(\theta)) \end{aligned}$$

## Proof that $\eta_{X,F}$ is natural in $X$ :

Given  $Y \xrightarrow{f} X$  in  $\mathbf{C}$ , have to show that

$$\begin{array}{ccc} \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) & \xrightarrow{\eta_{X,F}} & F(X) \\ (\mathcal{Y}f)^* \downarrow & & \downarrow F(f) \\ \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}Y, F) & \xrightarrow{\eta_{Y,F}} & F(Y) \end{array}$$

commutes in  $\mathbf{Set}$ . For all  $\mathcal{Y}X \xrightarrow{\theta} F$  we have

$$\begin{aligned} F(f)((\eta_{X,F}(\theta))) &\triangleq F(f)(\theta_X(\text{id}_X)) \\ &= \theta_Y(f^*(\text{id}_X)) && \text{by naturality of } \theta \\ &= \theta_Y(f) \\ &= \theta_Y(f_*(\text{id}_Y)) \\ &\triangleq (\theta \circ \mathcal{Y}f)_Y(\text{id}_Y) \\ &\triangleq \eta_{Y,F}(\theta \circ \mathcal{Y}f) \\ &\triangleq \eta_{Y,F}((\mathcal{Y}f)^*(\theta)) \end{aligned}$$



## Corollary of the Yoneda Lemma:

the functor  $\mathcal{Y} : \mathbf{C} \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is **full** and **faithful**.

In general, a functor  $F : \mathbf{C} \rightarrow \mathbf{D}$  is

- ▶ **faithful** if for all  $X, Y \in \mathbf{C}$  the function

$$\begin{array}{ccc} \mathbf{C}(X, Y) & \rightarrow & \mathbf{D}(F(X), F(Y)) \\ f & \mapsto & F(f) \end{array}$$

is injective:

$$\forall f, f' \in \mathbf{C}(X, Y), F(f) = F(f') \Rightarrow f = f'$$

- ▶ **full** if the above functions are all surjective:

$$\forall g \in \mathbf{D}(F(X), F(Y)), \exists f \in \mathbf{C}(X, Y), F(f) = g$$

## Corollary of the Yoneda Lemma:

the functor  $\mathcal{Y} : \mathbf{C} \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is **full** and **faithful**.

**Proof.** From the proof of the Yoneda Lemma, for each  $F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  we have a bijection

$$F(X) \xrightarrow{(\eta_{X,F})^{-1}} \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F)$$

By definition of  $(\eta_{X,F})^{-1}$ , when  $F = \mathcal{Y}Y$  the above function is equal to

$$\begin{aligned} \mathcal{Y}Y(X) = \mathbf{C}(X, Y) &\rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, \mathcal{Y}Y) \\ f &\mapsto f_* = \mathcal{Y}f \end{aligned}$$

So, being a bijection,  $f \mapsto \mathcal{Y}f$  is both injective and surjective; so  $\mathcal{Y}$  is both faithful and full. □

Recall (for a small category  $\mathbf{C}$ ):

Yoneda functor  $\mathcal{Y} : \mathbf{C} \rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}$

Yoneda Lemma: there is a bijection

$\mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, F) \cong F(X)$  which is natural both in  $F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  and  $X \in \mathbf{C}$ .

An application of the Yoneda Lemma:

**Theorem.** For each small category  $\mathbf{C}$ , the category  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  of presheaves is cartesian closed.

**Theorem.** For each small category  $\mathbf{C}$ , the category  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  of presheaves is cartesian closed.

**Theorem.** For each small category  $\mathbf{C}$ , the category  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  of presheaves is cartesian closed.

**Proof sketch.**

Terminal object in  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is the functor  $1 : \mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$  given by

$$\begin{cases} 1(X) \triangleq \{0\} \\ 1(f) \triangleq \text{id}_{\{0\}} \end{cases} \quad \text{terminal object in } \mathbf{Set}$$

**Theorem.** For each small category  $\mathbf{C}$ , the category  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  of presheaves is cartesian closed.

**Proof sketch.**

Product of  $F, G \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  is the functor  $F \times G : \mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$  given by

$$\begin{cases} (F \times G)(X) \triangleq F(X) \times G(X) & \text{cartesian product of sets} \\ (F \times G)(f) \triangleq F(f) \times G(f) \end{cases}$$

with projection morphisms  $F \xleftarrow{\pi_1} F \times G \xrightarrow{\pi_2} G$  given by the natural transformations whose components at  $X \in \mathbf{C}$  are the projection functions  $F(X) \xleftarrow{\pi_1} F(X) \times G(X) \xrightarrow{\pi_2} G(X)$ .

**Theorem.** For each small category  $\mathbf{C}$ , the category  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  of presheaves is cartesian closed.

**Proof sketch.**

We can work out what the value of the exponential  $G^F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  at  $X \in \mathbf{C}$  has to be using the Yoneda Lemma:

$$G^F(X) \cong \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, G^F) \cong \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X \times F, G)$$

Yoneda Lemma

universal property of  
the exponential

**Theorem.** For each small category  $\mathbf{C}$ , the category  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  of presheaves is cartesian closed.

**Proof sketch.**

We can work out what the value of the exponential  $G^F \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$  at  $X \in \mathbf{C}$  has to be using the Yoneda Lemma:

$$G^F(X) \cong \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X, G^F) \cong \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X \times F, G)$$

We take the set  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X \times F, G)$  to be the definition of the value of  $G^F$  at  $X$ ...



## Exponential objects in $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$ :

$$G^F(X) \triangleq \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X \times F, G)$$

Given  $Y \xrightarrow{f} X$  in  $\mathbf{C}$ , we have  $\mathcal{Y}Y \xrightarrow{\mathcal{Y}f} \mathcal{Y}X$  in  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  and hence

$$\begin{aligned} G^F(Y) \triangleq \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}Y \times F, G) &\rightarrow \mathbf{Set}^{\mathbf{C}^{\text{op}}}(\mathcal{Y}X \times F, G) \triangleq G^F(X) \\ \theta &\mapsto \theta \circ (\mathcal{Y}f \times \text{id}_F) \end{aligned}$$

We define

$$G^F(f) \triangleq (\mathcal{Y}f \times \text{id}_F)^*$$

Have to **check** that these definitions make  $G^F$  into a functor  $\mathbf{C}^{\text{op}} \rightarrow \mathbf{Set}$ .

## Application morphisms in $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$ :

Given  $F, G \in \mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , the application morphism

$$\text{app} : G^F \times F \rightarrow G$$

is the natural transformation whose component at  $X \in \mathbf{C}$  is the function

$$(G^F \times F)(X) \triangleq G^F(X) \times F(X) \triangleq \mathbf{Set}^{\mathbf{C}^{\text{op}}}(X \times F, G) \times F(X) \xrightarrow{\text{app}_X} G(X)$$

defined by

$$\text{app}_X(\theta, x) \triangleq \theta_X(\text{id}_X, x)$$

Have to **check** that this is natural in  $X$ .

## Currying operation in $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$ :

$$\left( H \times F \xrightarrow{\theta} G \right) \mapsto \left( H \xrightarrow{\text{cur } \theta} G^F \right)$$

Given  $H \times F \xrightarrow{\theta} G$  in  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$ , the component of  $\text{cur } \theta$  at  $X \in \mathbf{C}$

$$H(X) \xrightarrow{(\text{cur } \theta)_X} G^F(X) \triangleq \mathbf{Set}^{\mathbf{C}^{\text{op}}}( \multimap X \times F, G )$$

is the function mapping each  $z \in H(X)$  to the natural transformation  $\multimap X \times F \rightarrow G$  whose component at  $Y \in \mathbf{C}$  is the function

$$(\multimap X \times F)(Y) \triangleq \mathbf{C}(Y, X) \times F(Y) \rightarrow G(Y)$$

defined by

$$\boxed{((\text{cur } \theta)_X(z))_Y(f, y) \triangleq \theta_Y(H(f)(z), y)}$$

## Currying operation in $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$ :

$$\left( H \times F \xrightarrow{\theta} G \right) \mapsto \left( H \xrightarrow{\text{cur } \theta} G^F \right)$$

$$\left( (\text{cur } \theta)_X(z) \right)_Y(f, y) \triangleq \theta_Y(H(f)(z), y)$$

Have to **check** that this is natural in  $Y$ ,

then that  $(\text{cur } \theta)_X$  is natural in  $X$ ,

then that  $\text{cur } \theta$  is the unique morphism  $H \xrightarrow{\varphi} G^F$  in  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  satisfying  $\text{app} \circ (\varphi \times \text{id}_F) = \theta$ .

**Theorem.** For each small category  $\mathbf{C}$ , the category  $\mathbf{Set}^{\mathbf{C}^{\text{op}}}$  of presheaves is cartesian closed.

So we can interpret simply typed lambda calculus in any presheaf category.

More than that, presheaf categories (usefully) model dependently-typed languages.

## Lecture 16

# Monads

Used in Haskell to abstract generic aspects of computation (return a value, sequencing) and to encapsulate effectful code.

Concept imported into functional programming from category theory, first for its denotational semantics by Moggi and then for its practice by Wadler.

# Monads

Used in Haskell to abstract generic aspects of computation (return a value, sequencing) and to encapsulate effectful code.

Concept imported into functional programming from category theory, first for its denotational semantics by Moggi and then for its practice by Wadler.

Here, a quick overview of:

- ▶ Moggi's computational  $\lambda$ -calculus and its categorical semantics using (strong) monads
- ▶ monads and adjunctions



# Computational Lambda Calculus (CLC)

CLC extends STLC with new types, terms and equations...

**Types:**  $A, B, \dots ::=$  STLC types, plus

$T(A)$  type of “computations” of values of type  $A$

**Terms:**  $s, t, \dots ::=$  STLC terms, plus

$\text{return } t$  trivial computation

$\text{do}\{x \leftarrow s; t\}$  sequenced computation (**binds** free  $x$  in  $t$ )

As for STLC, we identify CLC syntax trees up to  $\alpha$ -equivalence, where  $=_{\alpha}$  is extended by the rules

$$\frac{t =_{\alpha} t'}{\text{return } t =_{\alpha} \text{return } t'} \text{ and } \frac{s =_{\alpha} s' \quad (y \ x) \cdot t =_{\alpha} (y \ x') \cdot t' \quad y \text{ does not occur in } \{s, s', x, x', t, t'\}}{\text{do}\{x \leftarrow s; t\} =_{\alpha} \text{do}\{x' \leftarrow s'; t'\}}$$

# Computational Lambda Calculus (CLC)

CLC extends STLC with new types, terms and equations...

**Types:**  $A, B, \dots ::=$  STLC types, plus

$T(A)$  type of “computations” of values of type  $A$

**Terms:**  $s, t, \dots ::=$  STLC terms, plus

$\text{return } t$  trivial computation

$\text{do}\{x \leftarrow s; t\}$  sequenced computation (binds free  $x$  in  $t$ )

**Typing rules:**

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \text{return } t : T(A)} \text{ (VAL)} \quad \frac{\Gamma \vdash s : T(A) \quad \Gamma, x : A \vdash t : T(B)}{\Gamma \vdash \text{do}\{x \leftarrow s; t\} : T(B)} \text{ (SEQ)}$$

**Equations...**

# CLC equations

Extend STLC  $\beta\eta$ -equality ( $\Gamma \vdash s =_{\beta\eta} t : A$ ) to a relation  $\Gamma \vdash s = t : A$  by adding the following rules:

$$\frac{\Gamma \vdash s : A \quad \Gamma, x : A \vdash t : T(B)}{\Gamma \vdash \text{do}\{x \leftarrow \text{return } s; t\} = t[s/x] : T(B)}$$

$$\frac{\Gamma \vdash t : T(A)}{\Gamma \vdash t = \text{do}\{x \leftarrow t; \text{return } x\} : T(A)}$$

$$\frac{\Gamma \vdash s : T(A) \quad \Gamma, x : A \vdash t : T(B) \quad \Gamma, y : B \vdash u : T(C)}{\Gamma \vdash \text{do}\{y \leftarrow \text{do}\{x \leftarrow s; t\}; u\} = \text{do}\{x \leftarrow s; \text{do}\{y \leftarrow t; u\}\}}$$

# CLC equations

Extend STLC  $\beta\eta$ -equality ( $\Gamma \vdash s =_{\beta\eta} t : A$ ) to a relation  $\Gamma \vdash s = t : A$  by adding the following rules:

$$\frac{\Gamma \vdash s : A \quad \Gamma, x : A \vdash t : T(B)}{\Gamma \vdash \text{do}\{x \leftarrow \text{return } s; t\} = t[s/x] : T(B)}$$

$$\frac{\Gamma \vdash t : T(A)}{\Gamma \vdash t = \text{do}\{x \leftarrow t; \text{return } x\} : T(A)}$$

$$\frac{\Gamma \vdash s : T(A) \quad \Gamma, x : A \vdash t : T(B) \quad \Gamma, y : B \vdash u : T(C)}{\Gamma \vdash \text{do}\{y \leftarrow \text{do}\{x \leftarrow s; t\}; u\} = \text{do}\{x \leftarrow s; \text{do}\{y \leftarrow t; u\}\}}$$

(To describe a particular notion of computation (I/O, mutable state, exceptions, concurrent processes, ...) one can consider extensions of vanilla CLC, e.g. with extra ground types, constants and equations.)

# Parameterised Kleisli triple

is the following extra structure on a category  $\mathbf{C}$  with binary products:

- ▶ a function mapping each  $X \in \text{obj } \mathbf{C}$  to an object  $T(X) \in \text{obj } \mathbf{C}$
- ▶ for each  $X \in \text{obj } \mathbf{C}$ , a  $\mathbf{C}$ -morphism  $X \xrightarrow{\eta_X} T(X)$
- ▶ for each  $\mathbf{C}$ -morphism  $X \times Y \xrightarrow{f} T(Z)$  a  $\mathbf{C}$ -morphism  $X \times T(Y) \xrightarrow{f^*} T(Z)$

satisfying...

# Parameterised Kleisli triple[cont.]

- ▶ if  $X \xrightarrow{f} X'$  and  $X' \times Y \xrightarrow{g} T(Z)$ , then

$$(g \circ (f \times \text{id}_Y))^* = g^* \circ (f \times \text{id}_{T(Y)})$$

- ▶ if  $X \times Y \xrightarrow{f} T(Z)$ , then

$$f^* \circ (\text{id}_X \times \eta_Y) = f$$

- ▶ if  $X \times Y \xrightarrow{f} T(Z)$  and  $X \times Z \xrightarrow{g} T(W)$ , then

$$(g^* \circ \langle \pi_1, f \rangle)^* = g^* \circ \langle \pi_1, f^* \rangle$$

# Examples in Set

**State:** fix a set  $S$  (of “states”) and define

$$T(X) \triangleq (X \times S)^S$$

$$\eta_X x s \triangleq (x, s)$$

$$f^*(x, t) s \triangleq f(x, y) s' \text{ where } t s = (y, s')$$

# Examples in Set

**State:** fix a set  $S$  (of “states”) and define

$$T(X) \triangleq (X \times S)^S$$

$$\eta_X x s \triangleq (x, s)$$

$$f^*(x, t) s \triangleq f(x, y) s' \text{ where } t s = (y, s')$$

computations are functions  $S \rightarrow X \times S$   
taking states to values in  $X$  paired with  
a next state

$f^*(x, \_)$  first “runs”  $t \in T(Y)$  in state  $s$  to get  $(y, s')$ ,  
then runs  $f(x, y) \in T(Z)$  in the new state  $s'$



# Examples in Set

**Error:**

$$T(X) \triangleq X + 1 = \{(0, x) \mid x \in X\} \cup \{(1, 0)\}$$

$$\eta_X x \triangleq (0, x)$$

$$f^*(x, t) \triangleq \begin{cases} f(x, y) & \text{if } t = (0, y) \\ (1, 0) & \text{if } t = (1, 0) \end{cases}$$

# Examples in Set

**Error:**

$$T(X) \triangleq X + 1 = \{(0, x) \mid x \in X\} \cup \{(1, 0)\}$$

$$\eta_X x \triangleq (0, x)$$

$$f^*(x, t) \triangleq \begin{cases} f(x, y) & \text{if } t = (0, y) \\ (1, 0) & \text{if } t = (1, 0) \end{cases}$$

computations are either copies  $(0, x)$  of values in  $x \in X$  or an error  $(1, 0)$

if  $t \in T(Y)$  is the error, then  $f^*(x, \_)$  propagates it, otherwise it acts like  $f$

# Examples in Set

**Continuations:** fix a set  $R$  (of “results”) and define

$$T(X) \triangleq R^{(R^X)}$$

$$\eta_X x \triangleq \lambda c \in R^X. c x$$

$$f^*(x, r) \triangleq \lambda c \in R^Z. r(\lambda y \in Y. f(x, y) c)$$

# Examples in Set

**Continuations:** fix a set  $R$  (of “results”) and define

$$T(X) \triangleq R^{(R^X)}$$

$$\eta_X x \triangleq \lambda c \in R^X. c x$$

$$f^*(x, r) \triangleq \lambda c \in R^Z. r(\lambda y \in Y. f(x, y) c)$$

computations are functions  $r : R^X \rightarrow R$   
mapping continuations  $c \in R^X$  of the  
computation to results  $r c \in R$

$f^*$  maps a computation  $r \in R^{(R^Y)}$  to the  
function taking a continuation  $c \in R^Z$  to  
the result of applying  $r$  to the  
continuation  $\lambda y \in Y. f(x, y) c$  in  $R^Y$

# Semantics of CLC

Given a ccc  $\mathbf{C}$  equipped with a parameterised Kleisli triple  $(T, \eta, (\_)*)$ , we can extend the semantics of STLC to one for CLC.

**Computation types:**  $\llbracket T(A) \rrbracket = T(\llbracket A \rrbracket)$

**Trivial computations:**

$$\llbracket \Gamma \vdash \text{return } t : T(A) \rrbracket = \llbracket \Gamma \rrbracket \xrightarrow{\llbracket \Gamma \vdash t : A \rrbracket} \llbracket A \rrbracket \xrightarrow{\eta_{\llbracket A \rrbracket}} T(\llbracket A \rrbracket)$$

**Sequencing:**  $\llbracket \Gamma \vdash \text{do}\{x \leftarrow s; t\} : T(B) \rrbracket = f^* \circ \langle \text{id}_{\llbracket \Gamma \rrbracket}, g \rangle$

$$\text{where } \begin{cases} f &= \llbracket \Gamma \rrbracket \times \llbracket A \rrbracket \xrightarrow{\llbracket \Gamma, x:A \vdash t : T(B) \rrbracket} T(\llbracket B \rrbracket) \\ g &= \llbracket \Gamma \rrbracket \xrightarrow{\llbracket \Gamma \vdash s : T(A) \rrbracket} T(\llbracket A \rrbracket) \end{cases}$$

(and where  $A$  is uniquely determined from the proof of  $\Gamma \vdash \text{do}\{x \leftarrow s; t\} : T(B)$ )

# Semantics of CLC

Given a ccc  $\mathbf{C}$  equipped with a parameterised Kleisli triple  $(T, \eta, (\_)*)$ , we can extend the semantics of STLC to one for CLC.

As for STLC versus cccs,

- ▶ the semantics of CLC in cc+Kleisli categories is equationally sound and complete
- ▶ one can use CLC as an internal language for describing constructs in cc+Kleisli categories
- ▶ there is a correspondence between equational theories in CLC and cc+Kleisli categories

# Monads

A **monad** on a category  $\mathbf{C}$  is given by a functor  $T : \mathbf{C} \rightarrow \mathbf{C}$  and natural transformations  $\eta : \text{id} \rightarrow T$  and  $\mu : T \circ T \rightarrow T$  satisfying

$$\begin{array}{ccc} T & \xrightarrow{T\eta} & T \circ T \xleftarrow{\eta_T} T \\ & \searrow \text{id}_T & \downarrow \mu \swarrow \text{id}_T \\ & & T \end{array} \qquad \begin{array}{ccc} T \circ T \circ T & \xrightarrow{\mu_T} & T \circ T \\ T\mu \downarrow & & \downarrow \mu \\ T \circ T & \xrightarrow{\mu} & T \end{array}$$

# Monads

A **monad** on a category  $\mathbf{C}$  is given by a functor  $T : \mathbf{C} \rightarrow \mathbf{C}$  and natural transformations  $\eta : \text{id} \rightarrow T$  and  $\mu : T \circ T \rightarrow T$  satisfying

$$\begin{array}{ccc} T & \xrightarrow{T\eta} & T \circ T \xleftarrow{\eta_T} T \\ & \searrow \text{id}_T & \downarrow \mu \swarrow \text{id}_T \\ & & T \end{array} \qquad \begin{array}{ccc} T \circ T \circ T & \xrightarrow{\mu_T} & T \circ T \\ T\mu \downarrow & & \downarrow \mu \\ T \circ T & \xrightarrow{\mu} & T \end{array}$$

If  $\mathbf{C}$  has binary products, then the monad is **strong** if there is a family of  $\mathbf{C}$ -morphisms  $(X \times T(Y) \xrightarrow{s_{X,Y}} T(X \times Y) \mid X, Y \in \text{obj } \mathbf{C})$  satisfying a number (7, in fact) of commutative diagrams (details omitted, see Moggi).



# Monads

A **monad** on a category  $\mathbf{C}$  is given by a functor  $T : \mathbf{C} \rightarrow \mathbf{C}$  and natural transformations  $\eta : \text{id} \rightarrow T$  and  $\mu : T \circ T \rightarrow T$  satisfying

$$\begin{array}{ccc} T & \xrightarrow{T\eta} & T \circ T \xleftarrow{\eta_T} T \\ & \searrow \text{id}_T & \downarrow \mu \swarrow \text{id}_T \\ & & T \end{array} \qquad \begin{array}{ccc} T \circ T \circ T & \xrightarrow{\mu_T} & T \circ T \\ T\mu \downarrow & & \downarrow \mu \\ T \circ T & \xrightarrow{\mu} & T \end{array}$$

If  $\mathbf{C}$  has binary products, then the monad is **strong** if there is a family of  $\mathbf{C}$ -morphisms  $(X \times T(Y) \xrightarrow{s_{X,Y}} T(X \times Y) \mid X, Y \in \text{obj } \mathbf{C})$  satisfying a number (7, in fact) of commutative diagrams (details omitted, see Moggi).

**FACT:** for a given category with binary products, “parameterised Kleisli triple” and “strong monad” are equivalent notions – each gives rise to the other in a bijective fashion.

# Monads and adjunctions

► Given an adjunction  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D} \quad \underline{F \dashv G}$

we get a monad  $(G \circ F, \eta, \mu)$  on  $\mathbf{C}$

$$\text{where } \begin{cases} \eta_X &= \overline{\text{id}_{FX}} \\ \mu_X &= G(\overline{\text{id}_{G(FX)}}) \end{cases}$$

E.g. for  $\mathbf{Set} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{U} \end{array} \mathbf{Mon}$  where  $U$  is the forgetful functor,  $T = U \circ F$  is

the **list monad** on  $\mathbf{Set}$  ( $T(X) = \mathbf{List} X$ ,  $\eta$  given by singleton lists,  $\mu$  by flattening lists of lists). It's a strong monad (all monads of  $\mathbf{Set}$  have a strength), but in general the monad associated with an adjunction may not be strong.

# Monads and adjunctions

► Given an adjunction  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D} \quad \underline{F \dashv G}$

we get a monad  $(G \circ F, \eta, \mu)$  on  $\mathbf{C}$

► Given a monad  $(T, \eta, \mu)$  on  $\mathbf{C}$  we get an adjunction

$$\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{C}^T \quad \underline{F \dashv G}$$

# Monads and adjunctions

$\mathbf{C}^T$  is the category of **Eilenberg-Moore algebras** for the monad  $T$ , which has objects  $(A, \alpha)$  with  $\alpha : T(A) \rightarrow A$  satisfying

- Given an adjunction

we get a monad  $(T, \eta, \mu)$

- Given a monad  $(T, \eta, \mu)$

$$\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{C}^T$$

$$\begin{array}{ccc} A & \xrightarrow{\eta_A} & T A \\ & \searrow \text{id}_A & \downarrow \alpha \\ & & A \end{array} \quad \begin{array}{ccc} T(T A) & \xrightarrow{\mu_A} & T A \\ T \alpha \downarrow & & \downarrow \alpha \\ T A & \xrightarrow{\alpha} & A \end{array}$$

and morphisms  $f(A, \alpha) \rightarrow (B, \beta)$  with  $f : A \rightarrow B$  satisfying

$$\begin{array}{ccc} T A & \xrightarrow{T f} & T B \\ \alpha \downarrow & & \downarrow \beta \\ A & \xrightarrow{f} & B \end{array}$$

# Monads and adjunctions

- ▶ Given an adjunction  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D} \quad \underline{F \dashv G}$

we get a monad  $(G \circ F, \eta, \mu)$  on  $\mathbf{C}$

- ▶ Given a monad  $(T, \eta, \mu)$  on  $\mathbf{C}$  we get an adjunction

$$\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{C}^T \quad \underline{F \dashv G}$$

- ▶ Starting from  $\mathbf{C} \begin{array}{c} \xrightarrow{F} \\ \xleftarrow{G} \end{array} \mathbf{D} \quad F \dashv G$  and forming the monad

$T = G \circ F$ , there's an obvious functor  $K : \mathbf{D} \rightarrow \mathbf{C}^T$ .

**Monadicity Theorems** impose conditions on  $G : \mathbf{D} \rightarrow \mathbf{C}$  which ensure that  $K$  is an equivalence of categories. E.g. **Mon** is equivalent to the category of Eilenberg-Moore algebras for the list monad on **Set** (and similarly for any algebraic theory).

# Some current themes involving category theory in computer science

- ▶ semantics of effects & co-effects in programming languages  
(monads and comonads)
- ▶ homotopy type theory  
(higher-dimensional category theory)
- ▶ structural aspects of networks, quantum computation/protocols, ...  
(string diagrams for monoidal categories)