

Category Theory and Functional Programming

Day 1

1 October 2009

Welcome

- 1 Why categories?
- 2 Why functional programming
- 3 Why the combination
- 4 This course

Why categories?

There's a tiresome young man in Bay Shore.
When his fiancée cried, 'I adore
The beautiful sea',
He replied, 'I agree,
It's pretty, but what is it for?'

Morris Bishop

3/32

Why categories?

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

4/32

Why categories?

- Describe structure through their **effect** on other structure
- **Internal** (set theory) vs. **external** (category theory)
- “Abstract nonsense”
- General theory of *things* (“**objects**”) and their *relations* (“**morphisms**”)
- Applicable in a huge variety of contexts
- **Organizing principle**

5/32

Why functional programming

SQL, Lisp, and Haskell are the only programming languages that I've seen where one spends more time thinking than typing.

Philip Greenspun

6/32

Why the combination

- Category theory is a **theory of functions**
- and of **functions on functions**
- Functional programming treats **functions as first-class objects**
- Hence category theory and functional programming share a **common mind-set**
- (And advanced functional programming uses some advanced categorical concepts)

7/32

Organization

- Four days of lectures and exercises
- plus some self-study
- 1, 7, 21, 28 October
- Exercise sessions are too short to do all exercises
- so do some of them on your own (or in groups!)

8/32

People

Lecturers



René R. Hansen



Uli Fahrenberg

Organizers



René R. Hansen



Uli Fahrenberg



Hans Hüttel

9/32

How to pass this course

- Some of the exercises (marked with *) are for student presentation
- choose one, solve it, present solution to audience ⇒ **PASS**
- Presentation lasts approx. 10 minutes
- Check your presentation with René or me before

Categories, Diagrams, and Morphisms

- 5 Categories (Pierce 1.1, 1.2)
- 6 Examples
- 7 Diagrams and commutativity
- 8 Examples
- 9 Monos, epis, isos (Pierce 1.3)
- 10 A category of transition systems (Winskel-Nielsen (Models) 2.1)

Categories

Examples

Diagrams and commutativity

Examples

Monos, epis, isos

A category of transition systems

Categories

- **Objects**
- **Arrows**, AKA morphisms
- For each arrow f , a **domain** and a **co-domain**
- (hence write $f : A \rightarrow B$)
- **Composition** of compatible arrows: for $f : A \rightarrow B$ and $g : B \rightarrow C$, we have $f; g : A \rightarrow C$
- (usually write $g \circ f$ instead of $f; g$, bummer...)

- Composition is **associative**: $h \circ (g \circ f) = (h \circ g) \circ f$
- And for each object A there's an **identity arrow** id_A , such that $f \circ \text{id}_A = f$ and $\text{id}_B \circ f = f$ for all arrows $f : A \rightarrow B$

- *That's all folks*

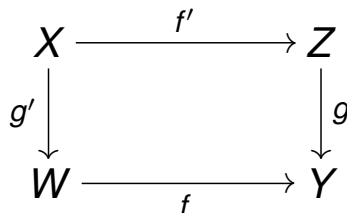
Examples of categories

Objects	Arrows
Sets	Functions
Groups	Homomorphisms
Monoids	Homomorphisms
Posets	Monotone functions
CPOs	Continuous functions
Graphs	Homomorphisms

13/32

Diagrams

- A diagram:



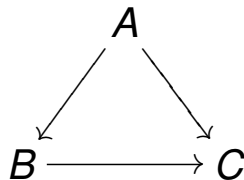
- so $f \circ g'$ and $g \circ f'$ exist
- The diagram **commutes** iff $f \circ g' = g \circ f'$

14/32

Comma categories

Given a category \mathcal{C} and an object $A \in \mathcal{C}$, define the **comma category** $A \downarrow \mathcal{C}$ by:

- Objects: $\mathcal{C}(A, B)$ for all $B \in \mathcal{C}$
 - all morphisms $f : A \rightarrow B$ in \mathcal{C} with **domain** A
- Arrows:



So the **objects** in $A \downarrow \mathcal{C}$ are **arrows** from \mathcal{C} , and the **arrows** in $A \downarrow \mathcal{C}$ are **commuting triangles** from \mathcal{C} !

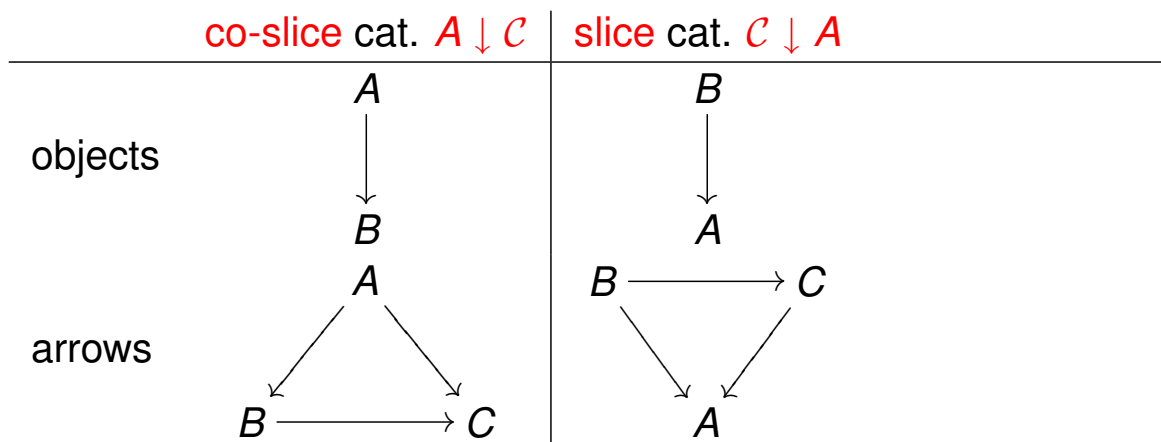
- And composition of arrows in $A \downarrow \mathcal{C}$ is composition of commuting triangles in \mathcal{C} .

This is called the **comma category**, or **co-slice** of \mathcal{C} under A .

15/32

Duality

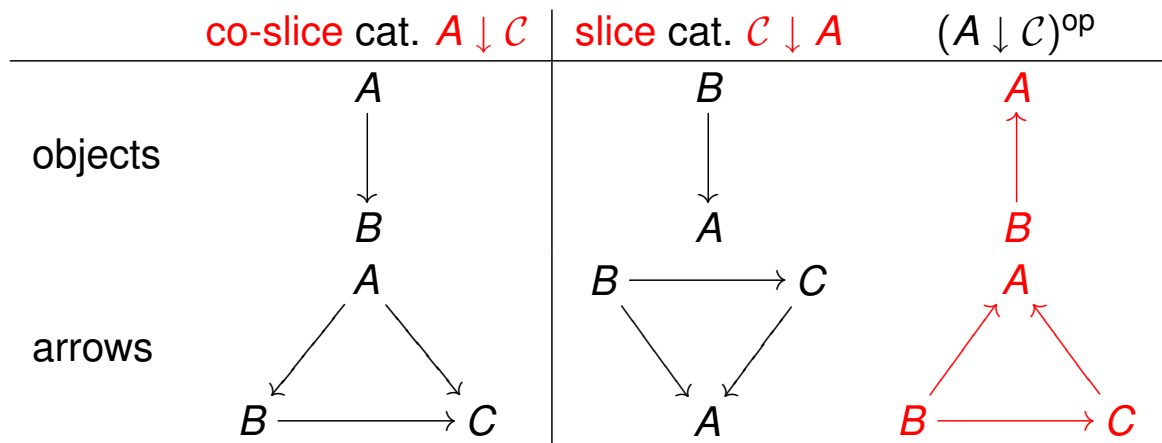
Where there's a **co-slice**, there's also a **slice** (for any object $A \in \mathcal{C}$):



16/32

Duality

Where there's a **co-slice**, there's also a **slice** (for any object $A \in \mathcal{C}$):



- So the slice is just the co-slice with **all arrows turned around**
- Definition: The **dual** of a category \mathcal{C} is the category \mathcal{C}^{op} , which has **the same objects but all arrows turned around**.

17/32

Monoids and pre-orders as categories

- A **monoid** is a set with an operation which is associative and has a unit.
- A monoid is a **category with one object**.
- A **pre-order** is a set with a relation which is reflexive and transitive.
- (A poset is a pre-order in which the relation is also antisymmetric.)
- A pre-order is a **category with at most one morphism between any two objects**.

18/32

Isomorphisms

Definition: An arrow $f : A \rightarrow B$ in a category \mathcal{C} is an **iso(morphism)** if it has an **inverse**, i.e. an arrow $g : B \rightarrow A$ for which $g \circ f = \text{id}_A$ and $f \circ g = \text{id}_B$.

$$A \begin{array}{c} \xrightarrow{f} \\ \xleftarrow{g} \end{array} B$$

- One also writes $g = f^{-1}$.
- These are just the usual isomorphisms in your favourite categories.
- Definition: Objects $A, B \in \mathcal{C}$ are **isomorphic** if there is an isomorphism $f : A \rightarrow B$.
- Isomorphic objects are **indistinguishable** from the point of view of category theory
- (because their *external* properties are the same).

19/32

Monomorphisms

In the category of **sets and functions**,

- an arrow $f : B \rightarrow C$ is **injective (one-to-one)** if $f(x) = f(y)$ implies $x = y$ for all $x, y \in B$.
- Equivalent: $f : B \rightarrow C$ is injective if **$f \circ g = f \circ h$ implies $g = h$ for all $g, h : A \rightarrow B$ and all A .**

$$A \begin{array}{c} \xrightarrow{g} \\ \xrightarrow{h} \end{array} B \xrightarrow{f} C$$

- Arrow-only (*external*) property!

Definition: An arrow $f : B \rightarrow C$ in a category \mathcal{C} is a **mono(morphism)** if $f \circ g = f \circ h$ implies $g = h$ for all $g, h : A \rightarrow B$ and all $A \in \mathcal{C}$.

Warning: In a lot of categories, “injective” does not make sense, and even if it does, it may not be the same as “mono”.

20/32

Epimorphisms

Again in the category of **sets and functions**,

- an arrow $f : A \rightarrow B$ is **surjective (onto)** if $\forall y \in B \exists x \in A : f(x) = y$.
- Equivalent: $f : A \rightarrow B$ is surjective if **$g \circ f = h \circ f$ implies $g = h$ for all $g, h : B \rightarrow C$ and all C .**

$$A \xrightarrow{f} B \begin{matrix} \xrightarrow{g} \\ \xrightarrow{h} \end{matrix} C$$

- Arrow-only (*external*) property!

Definition: An arrow $f : A \rightarrow B$ in a category \mathcal{C} is an **epi(morphism)** if $g \circ f = h \circ f$ implies $g = h$ for all $g, h : B \rightarrow C$ and all $C \in \mathcal{C}$.

Warning: In a lot of categories, “surjective” does not make sense, and even if it does, it may not be the same as “epi”.

21 / 32

Example (Pierce 1.3.6)

In the category of **monoids and homomorphisms**, the inclusion function $i : \mathbb{N} \hookrightarrow \mathbb{Z}$ is

- injective,
- a mono,
- not surjective,
- **but also an epi!**

22 / 32

A category of transition systems

- A **transition system** is a tuple (S, i, L, Tr) with $Tr \subseteq S \times L \times S$.
- A **morphism** of transition systems $T = (S, i, L, Tr)$, $T' = (S', i', L', Tr')$ is a pair $f = (\sigma, \lambda) : T \rightarrow T'$ of functions $\sigma : S \rightarrow S'$, $\lambda : L \rightarrow L'$ for which $\sigma(i) = i'$ and

$$(s_1, a, s_2) \in Tr \quad \text{implies} \quad (\sigma(s_1), \lambda(a), \sigma(s_2)) \in Tr'$$

- (Almost like a graph homomorphism)
- **But wait:** We want to be able to map labels in L to “nothing” (so we can abstract away actions)
- So we need **partial functions** $\lambda : L \rightarrow L'_\perp$
- And if $\lambda(a) = \perp$ above, then we want the transition to **disappear**.

23/32

Idle transitions

(A transition system is a tuple (S, i, L, Tr) with $Tr \subseteq S \times L \times S$.)

- Second try: Introduce **idle transitions**:

$$Tr_\perp = Tr \cup \{(s, \perp, s) \mid s \in S\}$$

- *Now it works:* A **morphism** of transition systems $T = (S, i, L, Tr)$, $T' = (S', i', L', Tr')$ is a pair $f = (\sigma, \lambda) : T \rightarrow T'$ of functions $\sigma : S \rightarrow S'$, $\lambda : L \rightarrow L'_\perp$ for which $\sigma(i) = i'$ and

$$(s_1, a, s_2) \in Tr \quad \text{implies} \quad (\sigma(s_1), \lambda(a), \sigma(s_2)) \in Tr'_\perp$$

- Together these form a category.

And we shall have to say much more about this category later.

24/32

Functors

- 11 Functors (Pierce 2.1)
- 12 Example
- 13 The category of categories
- 14 Natural transformations (Pierce 2.3)
- 15 Example

Functors

Example

The category of categories

Natural transformations

Example

Functors

Going up one level: We've seen lots of different categories now. What about a **category of categories**?

- Objects: categories
- Arrows: **functors**

Definition: A **functor** from a category \mathcal{C} to a category \mathcal{D} consists of a function F on objects and a function F on arrows

$$\begin{array}{ccc} \mathcal{C} & & \mathcal{D} \\ A & \xrightarrow{F} & F(A) \\ f \downarrow & \xrightarrow{F} & \downarrow F(f) \\ B & \xrightarrow{F} & F(B) \end{array}$$

for which $F(\text{id}_A) = \text{id}_{F(A)}$ and $F(g \circ f) = F(g) \circ F(f)$.

- A bit like **graph homomorphisms**!

Example (Pierce 2.1.2)

- The **Kleene star** (or **List**) function from sets to sets:

$$S \mapsto S^* = \text{List}(S) = \{\text{words } s_1 s_2 \dots s_n \mid n \in \mathbb{N}, \text{ all } s_i \in S\}$$

- Turn this into a functor from the category of **sets and functions** to itself:

$$f : S \rightarrow T \quad \mapsto \quad f^* : S^* \rightarrow T^*$$

$$f^*(s_1 s_2 \dots s_n) = f(s_1) f(s_2) \dots f(s_n)$$

- Or, in other words,

$$\text{List}(f) = \lambda s_1 s_2 \dots s_n . f(s_1) f(s_2) \dots f(s_n)$$

27/32

Example (Pierce 2.1.3)

- Actually, S^* is a **monoid** for all sets S :
 - Strings can be concatenated,
 - concatenation is associative
 - and has unit ε (empty string).
- Is Kleene star a **functor from sets to monoids**?
- Yes, for f^* is a monoid homomorphism for all functions f .

28/32

The category of categories

Recall the category of categories:

- Objects: categories
- Arrows: functors
- What about **composition of arrows**?

Definition: For functors $F : \mathcal{C} \rightarrow \mathcal{D}$, $G : \mathcal{D} \rightarrow \mathcal{E}$, the composite functor $G \circ F : \mathcal{C} \rightarrow \mathcal{E}$ is defined by

$$\begin{aligned} (G \circ F)(A) &= G(F(A)) && \text{on objects} \\ (G \circ F)(f) &= G(F(f)) && \text{on arrows} \end{aligned}$$

- (Nothing surprising here)
- Associativity ✓
- Identity functors ✓

29/32

Natural transformations

Going up another level:

- 1 Categories
- 2 Functors: arrows between categories
- 3 What about **arrows between functors**?

The **functor category** $\mathcal{D}^{\mathcal{C}}$ (for \mathcal{C}, \mathcal{D} categories) has

- objects: functors
- arrows: **natural transformations**

30/32

Natural transformations

Definition: A **natural transformation** $\eta : F \rightarrow G$ between functors $F, G : \mathcal{C} \rightarrow \mathcal{D}$ is a function from \mathcal{C} -objects to \mathcal{D} -arrows, $A \mapsto \eta_A : F(A) \rightarrow G(A)$ such that the diagrams

$$\begin{array}{ccc} F(A) & \xrightarrow{\eta_A} & G(A) \\ F(f) \downarrow & & \downarrow G(f) \\ F(B) & \xrightarrow{\eta_B} & G(B) \end{array}$$

commute for all arrows $f : A \rightarrow B$ in \mathcal{C} .

Example (Pierce 2.3.3)

rev : the function which **reverses** lists

- *Polymorphic*: input is list of any type
- So for any set S , we have a function $rev_S : S^* \rightarrow S^*$
- (Remember the Kleene star functor List from sets to monoids.)
- So rev is a function from sets to monoid homomorphisms,

$$rev : S \mapsto rev_S : S^* \rightarrow S^*$$

- A natural transformation $rev : List \rightarrow List$?
- Yes indeed ✓