! Категории типов. Часть 4. Пределы, сопряжения, монады.

Итак, позади три вводные части обзора, в которых мы познакомились с категориями, функторами между ними, а также естественными преобразованиями для функторов. В данной же части мы увидим, как все эти понятия помогут нам разобраться, какие вообще есть инструменты для конструирования новых типов. Мы получим представление, откуда берутся *монады* (наконец то!) и выясним, почему их так сложно композировать.

Мотивация

В одной из предыдущих частей обзора был продемонстрирован функтор, связывающий две разные категории одного и того же объекта — категории типов $\star$. Только в начальной категории морфизмами выступали конструкторы типов `F[_]`, а в конечной — эндофункторы `Functor[F]`. Полученный функтор сохранял композицию конструкторов типов, позволяя из любых `Functor[F]` и `Functor[G]` создавать `Functor[G ∘ F]`. А это, в свою очередь, означает, что если имеются реализации эндофункторов для каких-либо *базовых* `F[_]`, у нас автоматически есть эндофункоры для абсолютно любых `F[_]`!

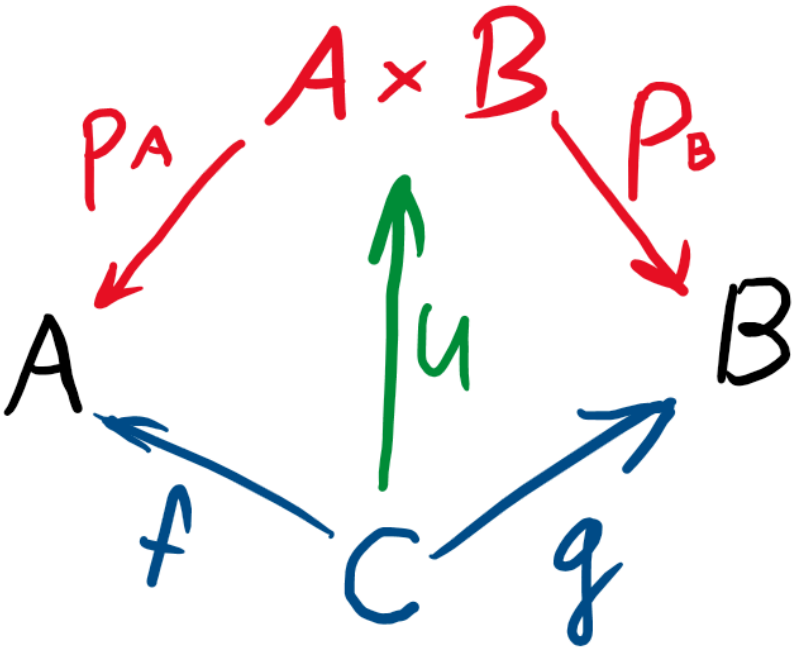
Открытым остаётся вопрос, какие конструкторы типов можно считать настолько «базовыми», что композируя их, можно получить вообще всё. Во многих публикациях данного цикла обзоров неоднократно упоминалось об алгебраической природе типов, о том, что в основе всего лежат элементарные математические операции — сумма, произведение и экпоненциал (а также понятия нуля и единицы). Однако, мне приходилось всячески избегать объяснения, *почему* происходит именно так. Дело в том, что ключ к понимаю этого лежит в теории категорий, так что только сейчас настало время разобраться с этим вопросом.

В первом обзоре цикла упоминалось, что любой тип можно определить через его *универсальное свойство*. Это понятие заимствовано из теории категории, с его помощью формулируются новые конструкции в категориях. Поэтому сперва давайте разберёмся, как это работает.

Универсальные свойства

В качестве примера напомню универсальное свойство произведения объектов:

> произведением двух объектов A и B выбранной категории называется такой объект $A \times B$ с морфизмами $p_A : A \times B \to A$ и $p_B : A \times B \to B$, что для любых объекта C и морфизмов $f : C \to A$ и $g : C \to B$ будет существовать единственный морфизм $u : C \to A \times B$, делающий такую диаграмму коммутирующей:



Определяемый объект $A \times B$, также как и все кандидаты C, имеют аналогичную пару морфизмов, идущих от них к A и B. Но среди всех кандидатов выделяется лишь один универсальный объект $A \times B$ — его *универсальный морфизм* u будет единственным *для любого* другого кандидата. Все остальные кандидаты будут либо сложнее (например, морфизм $A \times B \to A \times B \times X$ будет не единственным), либо диаграмма не будет коммутировать (по сути, любой кандидат, но выбранные для него морфизмы f и g не полиморфны по A, или B).

Определение любого универсального свойства в категории имеет такую структуру:

- обозначаются объекты, *через которые формулируется* универсальный (для произведения и суммы — это просто пара объектов A, B);
- выбирается семейство кандидатов с определёнными возможностями (в примере выше — пары морфизмов f, g или p_A, p_B);
- для каждого кандидата определяется морфизм u, связывающий его с целевым универсальным объектом;
- такой морфизм обязан быть единственным и обеспечивающий коммутируемость полученной диаграммы.

Как перечисленные пункты описать в терминах категории типов? «Возможности» кандидата — это некие преобразования, связывающие его с объектами подкатегории, фиксирующими формулировку универсального свойства. А требование коммутируеммости диаграммы очень напоминает условие *естественности* этого преобразования!

Конечно же, тут подразумевается некая функториальность для конструктора типов `F[_, ...]`. Его параметры даже могут иметь разную вариантность, поэтому в общем случае придётся иметь дело с *мультипрофункторами*, но в этой статье мы будем иметь дело, в первую очередь, с ковариантными функторами и бифункторами.

Конусы и ко-конусы

Давайте сперва посмотрим, как универсальные свойства выражаются с помощью естественных преобразований в некоторой категории $\mathcal{C}$. Объекты, через которые формулируется универсальный объект, образуют подкатегорию в $\mathcal{C}$. Для произведения объектов a и b такая подкатегория будет дискретной, содержащей только эти два объекта и их тождественные морфизмы. Но в случае других универсальных свойств устройство подкатегории может быть и сложнее.

Выбор конкретной подкатегории в $\mathcal{C}$ удобно выразить с помощью функтора, встраивающего в $\mathcal{C}$ некую модельную категорию $\mathcal{I}$. В случае универсального свойства произведения это и будет дискретная категория двух индексов, скажем, 1 и 2. Тогда функтор $F_{ab} : \mathcal{I} \to \mathcal{C}$ и будет отвечать за выбор конкретных a и b в $\mathcal{C}$.

Технически, сейчас можно было бы и обойтись без введения модельной категории и функтора из неё. Но кажется, что не значительно усложняя, мы только приобретём в наглядности рассуждений. Кроме того, такая точка зрения ещё пригодится нам в дальнейшем, поэтому полезно привыкнуть к ней заранее.

Ещё нам нужно выбрать объект-кандидат . Поручим это **константному функтору** $\Delta\, c : \mathcal{I} \to \mathcal{C}$. Естественное преобразование $\Delta\, c \rightsquigarrow F$ называется **конусом** функтора :

$$\mathrm{Cone}_F(c) : \Delta c \rightsquigarrow F$$

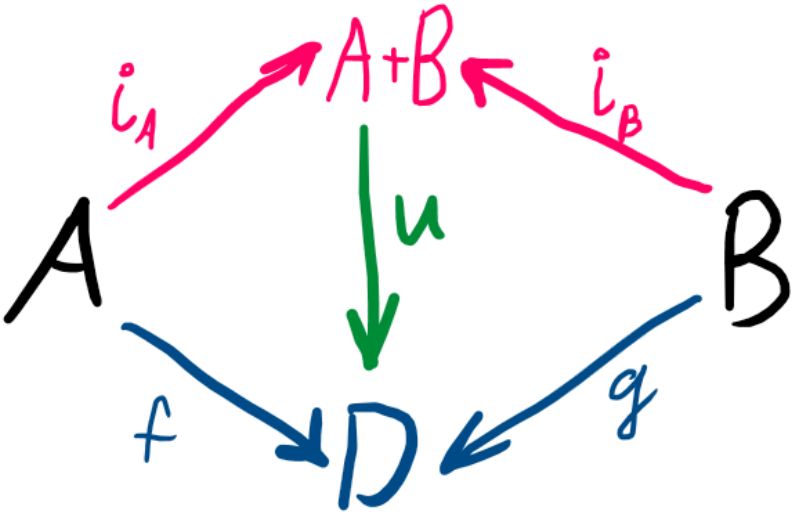
Естественность преобразования нужна для случаев, когда в модельной категории $\mathcal{I}$ есть другие морфизмы помимо тождественных (например, $1 \to 2$), и функтор F переносит их в также нетождественные морфизмы ($a \to b$). Тогда конус *обязан* быть естественным по отношению к этому функтору (в связке с константным, который погоды не делает).

Образ функтора F образует подкатегорию $F\mathcal{I}\ \ \mathcal{C}$, являющуюся «основанием» этого конуса, а объект $c : \mathcal{C}$ будет его вершиной:

!!! Рисунок конуса !!!

Для универсального свойства суммы объектов получим дуальную картину. Напомню,

> суммой двух объектов A и B выбранной категории называется такой объект $A + B$ с морфизмами $i_A : A \to A + B$ и $i_B : B \to A + B$, что для любых объекта D и морфизмов $f : A \to D$ и $g : B \to D$ будет существовать единственный морфизм $u : A + B \to D$, делающий такую диаграмму коммутирующей:



В сравнении с произведением, объекты на диаграмме сохранили своё положение, но все морфизмы оказались обращены. Кандидаты в сумму объектов описываются дуальной конструкцией, называемой **ко-конусом**:

$$\mathrm{Cocone}_F(c) : F \rightsquigarrow \Delta c$$

На диаграмме получаем как бы «перевёрнутый конус»:

!!! Рисунок ко-конуса !!!

Обратите внимание, что и моделирующая категория $\mathcal{I}$ и функтор F остались теми же, что и для произведения типов! Изменилось только направление естественного преобразования.

!! намёк на фундаментальность суммы и произведения

Кандидаты в любые универсальные объекты можно выразить либо через конусы, либо ко-конусы. Осталось только найти среди них *тот самый универсальный* (если он вообще есть).

Пределы и копределы

В формулировке универсального свойства упоминаются морфизмы, связывающие кандидатов и универсальные объекты так, что при этом не ломаются (ко)конусы («…делают диаграмму коммутирующей…»). Эти морфизмы образуют отношение частичного порядка среди кандидатов. Тогда может получиться так, что среди всех кандидатов выделяется «особенный», связанный с другими единственным (для каждого) универсальным морфизмом. Если такой объект существует (а это возможно далеко не всегда!), его (ко)конус, и, следовательно, он сам будут считаться «универсальными». Универсальный (ко)конус зависит лишь от функтора $F : \mathcal{I} \to \mathcal{C}$, он носит название **(ко)предел функтора** F и обозначается как $\mathrm{Lim}\, F$ ($\mathrm{Colim}\, F$).

!!! РИСУНОК ДЛЯ ПРЕДЕЛА

Итак, «…*для каждого* кандидата существует единственный универсальный морфизм...» Очевидно, что речь идёт о *взаимно-однозначном соответствии* кандидатов и универсальных морфизмов между ними и (ко)пределом. Причём, кандидаты определяются не столько объектами в категории $\mathcal{C}$, сколько связанными с ними конусами. Ведь формально универсальное свойство объекта представляет именно конус, который, будучи естественным преобразованием $\Delta c \rightsquigarrow F$, может иметь несколько различных реализаций. Все конусы, индексированные объектами категории $\mathcal{C}$ представляют собой функтор вида

$$\mathcal{F}_{cone} : \mathcal{C} \to \mathrm{Cone}_F(\_)$$

С другой стороны у нас обычные (типа, универсальные!) морфизмы $\mathrm{Hom}_{\mathcal{C}}(c, \mathrm{Lim}\, F)$. Они также индексированы объектами из $\mathcal{C}$ так, получается аналогичный функтор:

$$\mathcal{F}_{hom} : \mathcal{C} \to \mathrm{Hom}_{\mathcal{C}}(\_, \mathrm{Lim}\, F)$$

Несложно заметить, что оба функтора будут *контравариантными*. Область определения у них одна и та же — категория $\mathcal{C}$. А с областью значений всё немного интереснее. Дело в том, что конусы являются морфизмами в категории функторов, и дабы избежать парадоксов, принято считать, что все категории, с которыми мы работаем, являются *локально-малыми* — все морфизмы между двумя объектами *обязаны образовывать множество*. То есть, получается, что области значений обоих контравариантных функторов являются подкатегориями для дискретной категории Set, следовательно функторы будут одного и того же типа:

$$\mathcal{F}_{cone}, \mathcal{F}_{hom} : \mathcal{C} \to \mathrm{Set}$$

Для нас программистов картина не сильно поменяется, если в этом выражении заменить категорию множеств на дискретную же категорию значений (например, типа `Any`). Получим что-то вроде

$$\mathcal{F}_{cone}, \mathcal{F}_{hom} : \star \to Any$$

В любом случае, получается, что универсальное свойство задаётся *естественным изоморфизмом* этих функторов, то есть парой встречных естественных преобразований между ними, чьи композиции дают тождественные функторы:

$$\mathrm{Cone}_F \cong \mathrm{Hom}(\_, \mathrm{Lim}\, F)$$

Для копредела получаем дуальное выражение:

$$\mathrm{Cocone}_F \cong \mathrm{Hom}(\mathrm{Colim}\, F, \_)$$

!!! В спойлер! Предпучки

> Подобные контравариантные функторы (как правило, отображающие объекты категории во множество морфизмов), называются *предпучками*. Тогда естественные изоморфизмы выше — это *условия представимости* предпучка (ко)консуов в виде морфизмов в предел функтора. Подобная «представимость» имеет большое значение и мы с ней ещё столкнёмся в данном обзоре.
>
> Теория пучков сейчас активно развивается и «её инструментарий может оказаться очень полезным». Впрочем, этой фразой обычно и завершаются все упоминания предпучков))).
>
> Кстати, для того, чтобы называться полноценными «пучками», предпучкам не достаёт дополнительных ограничений… без которых программистам и так не плохо))).

Тут важно прочувствовать тонкий момент — слева у нас совокупность конусов, индексированная только функтором F, а справа — только его предел. Значит, это отношение можно считать уравнением для поиска предела функтора!

https://bartoszmilewski.com/2015/04/15/limits-and-colimits/
https://bartoszmilewski.com/2014/05/05/understanding-products-and-universal-constructions/
https://bartoszmilewski.com/2014/05/08/understanding-limits-2/

(Ко)пределы в категории типов

Как вообще можно построить новые типы? Исходно у нас нет возможности сделать что-то интересное с одним единственным типом, кроме как вернуть его же. Но можно попробовать скомбинировать пару существующих типов в какой-то новый. Получится функциональное отображение, на вход которому подаётся *два* типа, а на выходе получаем один новый. Нам то известно как «двухдырочные» конструкторы типов вида `(*, *) => *`. Их можно трактовать как функторы из произведения категорий типов в неё же: $\star \times \star \to \star$. Произведение двух категорий типов моделируется дискретной категорией $\mathcal{I} = 2$, состоящей из двух индексов 1 и 2, так что наши бифункторы можно записать как $F_{ab} : \star^2 \to \star$. Его конусами будут естественные преобразования с компонентами, индексированными объектами $\mathcal{I}$:

$$\begin{aligned}\mathrm{Cone}_{F_{ab}}(c)_1 &: c \to a,\\ \mathrm{Cone}_{F_{ab}}(c)_2 &: c \to b.\end{aligned}$$

Да, это просто пара проекторов типа-кандидата.

Вычислить универсальный конус честным способом не так уж и просто. Дело в том, что функторы из конечной категории $\mathcal{I}$ в категорию типов описываются зависимыми типами. Здесь мы не будем углубляться в эту тему.

Для дискретной $\mathcal{I}$ самый минимальный полиморфный универсальный объект обладает только теми возможностями, которые описываются компонентами конуса. Получается, что

$$\mathrm{Lim}\, F_{ab} = \forall i : \mathcal{I}.\ F_{ab}(i).$$

Это что-то вроде индексированного списка, у каждого элемента которого будет свой тип (~~зависимое произведение~~). В Scala самое близкое к такому типу — это кортеж, соответствующий произведению типов:

```
infix type ×[A, B] = (A, B)
```

Он действительно минимален — морфизмы из других кандидатов в него будет *единственным образом* «забывать» любую лишнюю информацию. А будучи полиморфными, проекторы предела будут факторизовать проекторы всех других кандидатов (сохранять коммутативность диаграммы).

Вообще, (ко)предел определяется не для категории индексов $\mathcal{I}$, а именно для функтора $\mathcal{I} \to \mathcal{C}$. Функторы бывают разные, в том числе, и Δ_c схлопывающий все морфизмы (не только тождественные!) в один. Однако, часто предел ищется только для наиболее «свободного» функтора, моделирующего категорию $\mathcal{I}$ в $\mathcal{C}$ без потерь. В этом случае, *(ко)предел определяется только структурой* $\mathcal{I}$. Например, очевидно, что если в дискретной $\mathcal{I}$ будет n объектов, то пределом функтора из неё в типы будет кортеж из n элементов.

Формально, можно вывести несколько разных вариантов предела функтора. Например, произведение типов можно описать с помощью кодирования Чёрча как `[A, B] =>> [C] => (A => B => C) => C`. Но все эти варианты будут изоморфны друг другу — предел функтора (если он существует) *единственен с точность до изоморфизма*.

Для того же бифунктора F_{ab} можно также определить и ко-конусы с такими компонентами:

$$\begin{aligned}\mathrm{Cocone}_{F_{ab}}(c)_1 &: a \to c,\\ \mathrm{Cocone}_{F_{ab}}(c)_2 &: b \to c.\end{aligned}$$

Очевидно, что это естественное преобразование для копредела задаёт «левый» и «правый» *конструкторы* суммы типов:

```
infix type +[A, B] = A Either B
```

В Scala наиболее подходящая реализация копредела в случае бóльших размеров категории $\mathcal{I}$ будет, пожалуй перечисление:

```
enum Sum3[A, B, C]:
  case One(a: A)
  case Two(b: B)
  case Tri(c: C)
```

И так, мы можем строить новые типы складывая и умножая существующие. А какие типы существуют *изначально* и как для них определяются универсальные свойства? Да совершенно аналогичным способом! Тогда как раньше мы строили новые типы из двух, моделируемых категорией $\mathcal{I} = 2$, теперь нам не требуется ни одного типа, что соответствует абсолютно пустой категории $\mathcal{I} = 0$! Единственно возможный функтор честно отображает пустую категорию в пустую диаграмму в $\star$, и от (ко)конусов остаются одни лишь вершины-кандидаты. Только для предела этого функтора все универсальные морфизмы будут вести *к нему*, в то время как для копредела они будут вести *от него* к кандидам. Это в точности соответствует универсальным свойствам *терминального и начального объектов* в категории типов — единичного (`Unit`) и нулевого (`Nothing`) типов!

!!! Предел для функции!!

!!! Непрерывность функтора - сохранение пределов

$$\operatorname{Lim}(F \circ G) = F \operatorname{Lim} G$$

- Непрерывность — это про "хорошее поведение" относительно ограничений и "обратных" конструкций.
- Правые сопряжённые (как (F)) — непрерывны, но не когерентны.

3. Почему сохранение пределов часто сводят к сохранению произведений?

- В категориях, где все пределы строятся из произведений и уравнителей (например, в категории множеств или типов), достаточно проверить сохранение именно этих двух видов пределов, чтобы гарантировать сохранение всех конечных пределов.
- Произведения — это "базовый" и самый простой вид пределов.
- Поэтому в практических примерах часто проверяют сохранение произведений как ключевой тест на сохранение пределов.
- Сохранение пределов — это сохранение всех универсальных конструкций пределов, включая произведения, уравнители и другие.
- Произведения — самый простой и базовый вид пределов, поэтому их сохранение часто рассматривают как важный и наглядный критерий.
- Но для полной проверки сохранения пределов нужно учитывать все виды пределов, а не только произведения.

!!! Когерентность функтора - сохранение копределов

- Когерентность — про "хорошее поведение" относительно объединний и "прямых" конструкций.
- Левые сопряжённые — когерентны, но не обязательно непрерывны.

> Функтор, предел которого даёт экспоненциал, действует из категории с двумя объектами и одним морфизмом (категория стрелки) в категорию типов.
> Универсальное свойство экспоненциала реализуется через изоморфизм между функциями X => (A => B) и (X, A) => B.

Сопряжения функторов

Whenever you see a natural isomorphism of hom sets, chances are there is an adjunction between two functors

The main difference between universal constructions and adjunctions is that the latter are defined globally — for all hom-sets

Монады

моноид и моноидальная категория.

Это два разных, хотя и связанных, понятия в теории категорий. Важно понимать разницу между ними, чтобы не путать их.

1. Моноид в теории категорий

• Где: Моноид определяется внутри некоторой моноидальной категории (обычно, но не всегда, это категория множеств Set с декартовым произведением в качестве тензорного произведения).
• Что: Это объект M вместе с двумя морфизмами:
• μ: M ⊗ M → M (операция "умножения" или "композиции").
• η: I → M (морфизм из единичного объекта в M, определяющий "единичный элемент").

где I — единичный объект моноидальной категории (например, одноэлементное множество {} в категории Set). ⊗ - это тензорное произведение в моноидальной категории.

• Свойства: Эти морфизмы должны удовлетворять двум аксиомам, выражающим ассоциативность и единичность:
• Ассоциативность: μ ∘ (μ ⊗ id_M) = μ ∘ (id_M ⊗ μ) : M ⊗ M ⊗ M → M
• Единица: μ ∘ (η ⊗ id_M) = λ : I ⊗ M → M и μ ∘ (id_M ⊗ η) = ρ : M ⊗ I → M (где λ и ρ - левый и правый унитары моноидальной категории).

• Пример (в категории множеств Set):
• M — множество (например, множество строк).
• μ — функция M × M → M (например, конкатенация строк).
• η — функция {} → M (выбирающая пустую строку).

Это обычный моноид, как мы его знаем в алгебре.

• Другие примеры:
• Моноид в категории эндофункторов — это монада.
• Группа — это моноид с дополнительным условием существования обратного элемента.

2. Моноидальная категория

• Что: Это сама категория с дополнительной структурой, а именно:
• Тензорное произведение: Бифунктор $\otimes : C \times C \to C$, который позволяет "умножать" два объекта категории C и получать новый объект той же категории.
• Единичный объект: Объект I в категории C, который является "единицей" для тензорного произведения.
• Ассоциатор: Естественный изоморфизм $\alpha_{A,B,C} : (A \otimes B) \otimes C \to A \otimes (B \otimes C)$, который выражает ассоциативность тензорного произведения (с точностью до изоморфизма).
• Левый и правый унуторы: Естественные изоморфизмы $\lambda_A : I \otimes A \to A$ и $\rho_A : A \otimes I \to A$, которые показывают, как единичный объект взаимодействует с тензорным произведением.

• Свойства: Ассоциатор и унуторы должны удовлетворять определённым аксиомам когерентности (например, пентагональная аксиома и треугольная аксиома), чтобы обеспечить согласованность операций.

• Примеры:
• Категория множеств Set с декартовым произведением × в качестве тензорного произведения и одноэлементным множеством {} в качестве единичного объекта.
• Категория векторных пространств Vect с тензорным произведением векторных пространств и полем k (рассматриваемым как 1-мерное векторное пространство) в качестве единичного объекта.
• Категория категорий Cat с декартовым произведением категорий и тривиальной категорией в качестве единичного объекта.

Главное отличие:

• Моноидальная категория — это тип категории, вид категории, которая обладает дополнительной структурой (тензорным произведением и единичным объектом). Она является фундаментом для определения моноидов.

• Моноид (в теории категорий) — это объект внутри уже существующей моноидальной категории, который удовлетворяет определённым условиям (ассоциативность и единичность). Он является конкретным элементом с определёнными свойствами.

Аналогия:

Представь себе:

• Моноидальная категория — это кухня. На кухне есть стол (тензорное произведение), место для хранения продуктов (объекты) и рецепты (правила когерентности).
• Моноид (в теории категорий) — это конкретный рецепт пирога (объект), который ты можешь приготовить на этой к

ухне. У пирога есть ингредиенты (элементы объекта) и способ их смешивания (операция умножения).

В итоге:

Моноидальная категория — это структура, в которой можно определять моноиды. Моноид — это конкретный объект, удовлетворяющий определённым условиям, внутри моноидальной категории.