Теория категорий

F-алгебры

Валерий Исаев

1 марта 2020 г.

План лекции

(Ко)индуктивные типы данных

Индуктивные типы данных

- Допустим мы хотим описать объект в произвольной категории, являющийся аналогом какой-либо структуры данных (списки, деревья, и так далее).
- Например, объект списков можно определить как бесконечную сумму

$$1 + A + A^2 + \ldots + A^n + \ldots$$

- Но это определение плохо обобщается на другие типы данных.

Индуктивные типы данных

- Проблема в том, что для произвольного типа данных сложно описать "объект данного типа фиксированного размера".
- Зато легко описать "объект размера $\leq$ данного".
- Например, если T_n – объект бинарных деревьев высоты $\leq n$, то $1 + A \times T_n \times T_n$ – объект бинарных деревьев высоты $\leq n + 1$.

Индуктивные типы данных

- Теперь, имея некоторый набор объектов $\{T_n\}_{n\in\mathbb{N}}$, нам нужно собрать из них один объект.
- В **Set** мы можем просто взять объединение множеств $\bigcup_{n\in\mathbb{N}} T_n$.
- В теории категорий объединение заменяется копределом.
- У нас есть очевидные морфизмы вложений $T_n \to T_{n+1}$. Теперь объект всех деревьев можно определить как следующий (секвенциальный) копредел:

$$T_0 \to T_1 \to T_2 \to \dots$$

Общее определение индуктивных типов данных

- Любой (ко)индуктивный тип данных можно задать в виде функтора $F : \mathbf{C} \to \mathbf{C}$.
- Функтор, соответствующий, спискам определяется как $L_A(X) = 1 + A \times X$.
- Функтор, соответствующий, бинарным деревьям определяется как $T_A(X) = 1 + A \times X \times X$.
- В общем случае $F_D(X)$ задается как правая часть определения типа данных D, в котором все рекурсивные вхождения этого типа заменены на X.
- Таким образом, если X является интерпретацией D, то он должен удовлетворять уравнению $X \simeq F(X)$.

Алгебры над эндофунктором

- Существует два канонических способа найти решение уравнения $X \simeq F(X)$, как начальную F-алгебру или конечную F-коалгебру.
- Если $F : \mathbf{C} \to \mathbf{C}$ – некоторый эндофунктор, то *F-алгебра* – это пара (X, α), где X – объект $\mathbf{C}$, а $\alpha : F(X) \to X$ – морфизм $\mathbf{C}$.
- Морфизм F-алгебр (X, α) и (Y, β) – это морфизм $f : X \to Y$ в $\mathbf{C}$ такой, что следующая диаграмма коммутирует:

$$\begin{tikzcd} F(X) \arrow[r, "\alpha"] \arrow[d, "F(f)"'] & X \arrow[d, "f"] \\ F(Y) \arrow[r, "\beta"'] & Y \end{tikzcd}$$

Начальные алгебры

- Легко видеть, что определения на предыдущем слайде задают категорию, которую мы будем обозначать $F\text{-}\mathrm{alg}$.
- Начальный объект этой категории называется начальной F-алгеброй, и если она существует, то она является решением уравнения $X \simeq F(X)$.
- Если функтор F сохраняет секвенциальные копределы, то начальную F-алгебру можно определить как копредел следующей диаграммы:

$$0 \xrightarrow{!_{F(0)}} F(0) \xrightarrow{F(!_{F(0)})} F(F(0)) \xrightarrow{F(F(!_{F(0)}))} F(F(F(0))) \to \ldots$$

Начальные алгебры и индуктивные типы

- Начальные F-алгебры можно использовать для интерпретации индуктивных типов данных.
- Если $N(X) = X \amalg 1$ – функтор, соответствующий типу данных унарных натуральных чисел, то начальная N-алгебра – это в точности объект натуральных чисел.
- Если $L_A(X) = 1 \amalg A \times X$ – функтор, соответствующий спискам, то начальная L_A-алгебра в **Set** – это множество конечных списков элементов A.
- В общем случае начальная L_A-алгебра – это копроизведение объектов 1, A, A^2, A^3, ...

Коалгебры над эндофунктором

- *Коалгебры* над $F : \mathbf{C} \to \mathbf{C}$ определяются дуальным образом как пары (X, α), где X – объект $\mathbf{C}$, а $\beta : X \to F(X)$ – морфизм $\mathbf{C}$.
- По дуальности конечные F-коалгебры тоже являются решением уравнения $X \simeq F(X)$.
- Если функтор F сохраняет секвенциальные пределы, то конечную F-коалгебру можно определить как предел следующей диаграммы:

$$\ldots \to F(F(F(1))) \xrightarrow{F(F(!_{F(1)}))} F(F(1)) \xrightarrow{F(!_{F(1)})} F(1) \xrightarrow{!_{F(1)}} 1$$

- Терминальные коалгебры являются интерпретацией коиндуктивных типов данных.
- Например, терминальная L_A-коалгебра в **Set** – это множество потенциально бесконечных списков.