Лямбда Исчисление

Автор Федоров И.И.

1 сентября 2017 г.

Содержание

1 Необходимые понятия

1.1 Машина Тьюринга

1.1.1 Абстрактный автомат

Абстрактный автомат (в теории алгоритмов) — математическая абстракция, модель дискретного устройства, имеющего один вход, один выход и в каждый момент времени находящегося в одном состоянии из множества возможных. На вход этому устройству поступают символы одного языка, на выходе оно выдаёт символы (в общем случае) другого языка.

Определение 1. Абстрактный автомат это пятёрка

$$M = (S, X, Y, \delta, \lambda)$$

- S — множество состояний автомата,
- X, Y — конечные входной и выходной алфавиты соответственно, из которых формируются строки, считываемые и выдаваемые автоматом,
- $\delta : S \times X \to S$ — функция переходов,
- $\lambda : S \times X \to Y$ — функция выходов.

Функционирование автомата в дискретные моменты времени t может быть описано системой рекуррентных соотношений:

$$s(t+1) = \delta(s(t), x(t))$$
$$y(t) = \lambda(s(t), x(t))$$

Определение 2. Абстрактный автомат с выделенным начальным состоянием называется инициальным автоматом. Таким образом абстрактный автомат определяет семейство инициальных автоматов $(s_i, A), s_i \in S$.

Определение 3. Если функции переходов и выходов однозначно определены для каждой пары $(s, x) \in S \times X$, то автомат называют детерминированным. В противном случае автомат называют недетерминированным или частично определенным. Если функция переходов и/или функция выходов являются случайными, то автомат называют вероятностным.

1.1.2 Конечный автомат

Определение 4. Конечный автомат — абстрактный автомат, число возможных внутренних состояний которого конечно.

Определение 5. Конечный автомат без выхода это пятёрка

$$M = (A, S, s_0, F, \delta)$$

- A — входной алфавит (конечное множество входных символов), из которого формируются входные слова, воспринимаемые конечным автоматом;
- S — множество внутренних состояний;
- s_0 — начальное состояние ($s_0 \in S$);
- F — множество допускающих состояний ($F \subset S$);
- δ — функция переходов, определенная как отображение $\delta\colon S \times A \to S$.

Принято полагать, что конечный автомат начинает работу в состоянии s_0, последовательно считывая по одному символу входного слова (цепочки входных символов). Считанный символ переводит автомат в новое состояние в соответствии с функцией переходов.

Определение 6. Будем говорить, что автомат допускает (англ. accept) слово, если после окончания описанного выше процесса автомат окажется в допускающем состоянии.

Замечание. Если в какой-то момент из текущего состояния нет перехода по считанному символу, то будем считать, что автомат не допускает данное слово. При реализации вместо отдельного рассмотрения данного случая иногда удобно вводить фиктивную нетерминальную «дьявольскую вершину» (также тупиковое состояние, сток), из которой любой переход ведет в неё же саму, и заменить все несуществующие переходы на переходы в «дьявольскую вершину»

Определение 7. Автоматы называются изоморфными (англ. isomorphic), если существует биекция между их вершинами такая, что сохраняются все переходы, терминальные состояния соответствуют терминальным, начальные — начальным.

Пример (Контролер нечетности единиц)**.** Задание распознающего автомата:

$$A = \{0, 1\}, S = \{Odd, Even\}, s_0 = Even, F = \{Odd\}$$

$$\delta(Even, 0) = Even$$

$$\delta(Even, 1) = Odd$$

$$\delta(Odd, 0) = Odd$$

$$\delta(Odd, 1) = Even$$

Задача 1. Пусть $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, убедиться что входная строка отсортирована по неубыванию.

Задача 2. Пусть $A =$ русский алфавит, убедиться что входная строка содержит подстроку "саша".

Задача 3. Пусть $A =$ латинский алфавит, убедиться что входная строка содержит подстроку "ababva".

1.1.3 Машина Тьюринга

Неформально машина Тьюринга определяется как устройство, состоящее из двух частей: бесконечной одномерной ленты, разделённой на ячейки, головки, которая представляет собой детерминированный конечный автомат. При запуске машины Тьюринга на ленте написано входное слово, причём на первом символе этого слова находится головка, а слева и справа от него записаны пустые символы. Каждый шаг головка может перезаписать символ под лентой и сместиться на одну ячейку, если автомат приходит в допускающее или отвергающее состояние, то работа машины Тьюринга завершается.

Определение 8. Формально машина Тьюринга (англ. Turing machine) определяется как кортеж из семи элементов $\langle A, a_0, S, Y, N, s_0, \delta \rangle$

- A — алфавит, из букв которого могут состоять входные слова,
- $a_0 \in A$ - пробельный символ(будем обозначать B),
- S — множество состояний управляющего автомата,
- $Y \in S$ — допускающее состояние автомата,
- $N \in S$ — отвергающее состояние автомата,
- $s_0 \in S$ — стартовое состояние автомата,
- $\delta : S \times A \to S \times A \times \{\leftarrow, \rightarrow, \downarrow\}$ — всюду определённая функция перехода автомата.

Определение 9. Назовём конфигурацией машины Тьюринга тройку (содержимое ленты, текущая позиция головки, текущее состояние машины).

Замечание. Отличие машин Тьюринга от конечных автоматов:

- Машины Тьюринга могут как записывать данные на ленту так и считывать данные с ленты.
- Головка может передвигаться налево и направо.
- Необязательно считывать входные данные целиком. С другой стороны, вычисление может продолжаться (даже бесконечно) после того, как все входные данные были считаны.
- При достижении допускающего или недопускающего состояния вычисление останавливается.

Определение 10. Машина Тьюринга M вычисляет функцию $f : B^* \to B^*$ (где B — подмножество алфавита машины, не содержащее пустого символа), если для каждого w из области определения функции f результат работы M равен $f(w)$, а для каждого w не из области определения f машина M не останавливается на входе w.

Пример. Для начала приведём пример машины-преобразователя, которая прибавляет единицу к числу, записанному на ленте в двоичной записи. Алгоритм следующий:

- в стартовом состоянии головка идёт вправо от старшего бита к младшему биту до первого пробельного символа, после чего переходит в состоянии L,
- в состоянии L головка идёт влево от младшего бита к старшему и заменяет все единицы на нули,
- встретив ноль или пробельный символ головка записывает единицу и переходит в допускающее состояние Y.

Формально: $A = \{0, 1, B\}, S = \{L, R\}$. Таблица функции δ приведена ниже:

	0	1	B
R	$\langle R, 0, \rightarrow \rangle$	$\langle R, 1, \rightarrow \rangle$	$\langle L, B, \leftarrow \rangle$
L	$\langle Y, 1, \downarrow \rangle$	$\langle L, 0, \leftarrow \rangle$	$\langle Y, 1, \downarrow \rangle$

Задача 4. Реалиализовать машину Тьюринга которая прибавляет единицы в десятичной системе счисления.

Задача 5. Покажите, что функция «обращение», переворачивающая слово задом наперед, вычислима на машине Тьюринга.

Задача 6. Реализуйте машину Тьюринга которая осуществляет сложение 2х чисел в двоичной системе счисления.

Теорема 1 (О существовании универсальной машины Тьюринга)**.** *Существует машина Тьюринга, которая может заменить собой любую машину Тьюринга. Получив на вход программу и входные данные, она вычисляет ответ, который вычислила бы по входным данным машина Тьюринга, чья программа была дана на вход.*

Задача 7. Реализовать универсальную машину Тьюринга.

Определение 11. Вычислимые функции — это множество функций вида $f\colon N \to N$, которые могут быть реализованы на машине Тьюринга. Задачу вычисления функции f называют алгоритмически разрешимой или алгоритмически неразрешимой, в зависимости от того, возможно ли написать алгоритм, вычисляющий эту функцию.

Пример. Любая функция с конечной областью определения вычислима.

1.1.4 Тезис Чёрча-Тьюринга

Тезис Чёрча—Тьюринга — фундаментальное утверждение для многих областей науки, таких, как теория вычислимости, информатика, теоретическая кибернетика и др. Это утверждение было высказано Алонзо Чёрчем и Аланом Тьюрингом в середине 1930-х годов.

В самой общей форме оно гласит, что любая интуитивно вычислимая функция является частично вычислимой, или, эквивалентно, может быть вычислена с помощью некоторой машины Тьюринга.

Тезис Чёрча—Тьюринга невозможно строго доказать или опровергнуть, поскольку он устанавливает «равенство» между строго формализованным понятием частично вычислимой функции и неформальным понятием «интуитивно вычислимой функции».

Физический тезис Чёрча—Тьюринга гласит: Любая функция, которая может быть вычислена физическим устройством, может быть вычислена машиной Тьюринга.

2 Чистое лямбда исчисление

2.1 Основные определения

2.1.1 Лямбда исчисление

Определение 12. Множество λ-термов Λ строится из переменных $V = \{x, y, z \ldots\}$ следующим образом:

$$x \in V \Rightarrow x \in \Lambda$$

$$M, N \in \Lambda \Rightarrow (M\ N) \in \Lambda$$

$$M \in \Lambda, x \in V \Rightarrow (\lambda x.M) \in \Lambda$$

Замечание. Общеприняты следующие соглашения:

- Аппликация λ-термов левоассоциативна, т.е. $a\ b\ c\ d = ((a\ b)\ c)\ d$.
- Абстракция забирает себе всё, до чего дотянется: $\lambda x.\lambda y.\lambda z.z\ y\ x \equiv \lambda x.(\lambda y.(\lambda z.((z\ y)\ x)))$
- Внешние скобки опускаются.
- Вместо $\lambda x_1.\lambda x_2 \ldots \lambda x_n.M$ пишут $\lambda x_1 \ldots\ x_n.M$.

Пример. Примеры λ-термов: x, $x\ y$, $x\ x$, $\lambda x.x$,$\lambda x.y$, $\lambda x.x\ y$

Определение 13. Множество свободных переменных $FV(T)$ в λ-терме T определяется следующим образом:

$$FV(x) = \{x\}$$

$$FV(M\ N) = FV(M) \cup FV(N)$$

$$FV(\lambda x.M) = FV(M) \setminus \{x\}$$

Определение 14. Множество связанных переменных $BV(T)$ в λ-терме T определяется следующим образом:

$$BV(x) = \emptyset$$

$$BV(M\ N) = BV(M) \cup BV(N)$$

$$BV(\lambda x.M) = BV(M) \cup \{x\}$$

Определение 15. M - замкнутый терм(или комбинатор), если $FV(M) = \emptyset$.

Пример. Примеры комбинаторов:

- $I = \lambda x.x$
- $K = \lambda x\ y.x$
- $K_* = \lambda x\ y.y$
- $S = \lambda f\ g\ x.f\ x\ (g\ x)$
- $B = \lambda f\ g\ x.f\ (g\ x)$

Определение 16. Определим операцию подстановки := следующим образом:

- $x[x := T] = T$
- $y[x := T] = y$, если $x \neq y$
- $(M_1\ M_2)[x := T] = M_1[x := T]\ M_2[x := T]$
- $(\lambda x.M)[x := T] = \lambda x.M$
- $(\lambda y.M)[x := T] = \lambda y.(M[x := T])$, если $x \neq y$ и либо $x \notin FV(M)$, либо $y \notin FV(T)$
- $(\lambda y.M)[x := T] = \lambda z.(M[y := z][x := T])$ в противном случае, причём $z \notin FV(M) \cup FV(T)$

Определение 17. α-конверсией(иногда α-эквивалентностью или даже α-редукцией) называется преобразование $\lambda x.A \to_\alpha \lambda y.A[x := y]$ (все свободные вхождения переменной x в A заменяются на y; при этом переменная y не должна входить в исходный терм A).

Определение 18. β-редукция называется преобразование $(\lambda x.A)\ B \to_\beta A[x := B]$ (все свободные вхождения переменной x в A заменяются на терм B(при этом предполагается, что при подстановке в A свободные переменные терма B не попадают в область действия квантора по одноименным переменным)).

Определение 19. η-редукция называется преобразование $(\lambda x.A\ x) \to_\eta A$, если $x \notin FV(A)$.

Определение 20. Термы A и B называются равными, если существует цепочка λ-термов

$$A \to C_1 \to C_2 \to \cdots \to C_n \to B$$

где каждый следующий терм получается из предыдущего либо с помощью α-конверсии, либо с помощью β-редукции, либо с помощью преобрзования обратного к β-редукции, либо с помощью η-редукции.

Определение 21. Говорят, что терм A находится в нормальной форме, если к нему нельзя применить β-редукцию.

Определение 22. Говорят, что у терма A есть нормальная форма, если существует эквивалентный терм в нормальной форме.

Задача 8. Придумать терм не имеющий нормальной формы.

Задача 9. Придумать терм который после применения β-редукции становится "больше".

2.1.2 Каррирование

Утверждение 1 (Шейнфинкель, 1924)**.** *Функция нескольких переменных может быть описана как конечная последовательность функций одной переменной.*

Доказательство. Пусть $\phi(x, y)$ терм содержащий свободные переменные x и y. Введём, путём последовательных абстракций

$$\Phi_x = \lambda y.\phi(x, y)$$

$$\Phi = \lambda x.\Phi_x = \lambda x.(\lambda y.\phi(x, y)) = \lambda x\ y.\phi(x, y)$$

Тогда применение этого Φ к произвольным X и Y может быть выполнена последовательно

$$\Phi\ X\ Y = (\Phi\ X)\ Y = \Phi_x\ Y = (\lambda y.\phi(X, y))\ Y = \phi(X, Y)$$

□

Определение 23. Каррирование (англ. carrying) — преобразование функции от многих переменных(функции от кортежа) в функцию, берущую свои аргументы по одному.

2.1.3 Свойства редукционных графов

Определение 24. Через $G_\beta(T)$, где T это терм, будем обозначать редукционный граф. Вершинами этого графа являются термы, а стрелками β-редукции.

Определение 25 (Свойство сильной нормализуемости(SN))**.** Если любой путь исходящий из терма T заканчивается, то говорят, что $G_\beta(T)$ обладает свойством сильной нормализуемости.

Определение 26 (Свойство слабой нормализуемости(WN))**.** Если существует путь приводящий терм T к нормально форме, то говорят, что $G_\beta(T)$ обладает свойством слабой нормализуемости.

Определение 27 (Свойство Чёрча-Россера(CR))**.** Если $A \twoheadrightarrow_\beta T_1$ и $A \twoheadrightarrow_\beta T_2$, а так же существует терм U такой, что $T_1 \twoheadrightarrow_\beta U$ и $T_2 \twoheadrightarrow_\beta U$, то говорят, что $G_\beta(A)$ обладает свойством Чёрча-Россера.

Определение 28 (Слабое свойство Чёрча-Россера(WCR))**.** Если $A \to_\beta T_1$ и $A \to_\beta T_2$, а так же существует терм U такой, что $T_1 \twoheadrightarrow_\beta U$ и $T_2 \twoheadrightarrow_\beta U$, то говорят, что $G_\beta(A)$ обладает слабым свойством Чёрча-Россера.

Теорема 2. *Для любого T:$G_\beta(T)$ обладает свойством CR.*

Доказательство. Введём вспомогательное исчисление $\to_l$

1. $\forall T : T \to_l T$
2. $T \to_l T' : \lambda x.T \to_l \lambda x.T'$
3. $T_1 \to_l T_1', T_2 \to_l T_2' : (T_1\ T_2) \to_l (T_1'\ T_2')$
4. $T_1 \to_l T_1', T_2 \to_l T_2' : (\lambda x.T_1)\ T_2 \to_l T_1'[x := T_2']$

Утверждение. $T \to_\beta T' \Rightarrow T \to_l T' \Rightarrow T \twoheadrightarrow_\beta T'$

Доказательство. Первая импликация следует из четвёртого правила, если подставить $T_1' = T_1, T_2' = T_2$. Последняя импликация доказывается индукцией по выводу $T \to_l T'$. □

Лемма 1 (О подстановке)**.** $x \neq y, x \notin FV(V) \Rightarrow U[x := W][y := V] =_\alpha U[y := V][x := W[y := V]]$

Доказательство. Очевидно. □

Лемма 2. $U \to_l U', V \to_l V' : U[x := V] \to_l U'[x := V']$

Доказательство. Индукция по выводу $U \to_l U'$

1. $U \to_l U', U = U'$ - индукция по построению U
   (a) $U = a$
      i. $a = x$, тогда $x[x := V] \to_l x[x := V'] = V \to_l V'$

ii. $a \neq x$, тогда $a[x := V] \to_l a[x := V'] = a \to_l a$

(b) $U = \lambda a.A$, тогда $A[x := V] \to_l A[x := V']$ и по правилу 2 $\lambda a.A[x := V] \to_l \lambda a.A[x := V']$

(c) $U = A_1\ A_2$, тогда $A_1[x := V] \to_l A_1[x := V']$ и $A_2[x := V] \to_l A_2[x := V']$, а по правилу 3 $(A_1\ A_2)[x := V] \to_l (A_1\ A_2)[x := V']$

2. $U_1 \to_l U_1' : \lambda y.U_1 \to_l \lambda y.U_1'$, где $U = \lambda y.U_1, U' = \lambda y.U_1'$, тогда можно применить предположение индукции сначала к $U_1 \to U_1'$ и получить $U_1[x := V] \to_l U_1'[x := V']$, а потом воспользоваться правилом 2, тогда $\lambda y.U_1[x := V] \to_l \lambda y.U_1'[x := V']$.

3. $U_1 \to_l U_1', U_2 \to_l U_2' : (U_1\ U_2) \to_l (U_1'\ U_2')$, где $U = (U_1\ U_2), U' = (U_1'\ U_2')$, тогда можно применить предположение индукции сначала к $U_1 \to U_1', U_2 \to U_2'$ и получить $U_1[x := V] \to_l U_1'[x := V'], U_2[x := V] \to_l U_2'[x := V']$, а потом воспользоваться правилом 3, тогда $(U_1\ U_2)[x := V] \to_l (U_1'\ U_2')[x := V']$.

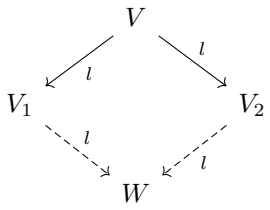
4. $U_1 \to_l U_1', U_2 \to_l U_2' : (\lambda y.U_1)\ U_2 \to_l U_1'[x := U_2']$, где $U = (\lambda y.U_1)\ U_2, U' = U_1'[x := U_2']$.

   Хотим $(\lambda y.U_1[x := V])\ U_2[x := V] \to_l U_1'[y := U_2'][x := V']$

   По предположению индукции $U_1[x := V] \to_l U_1'[x := V'], U_2[x := V] \to_l U_2'[x := V']$ и по правилу 4 $(\lambda y.U_1[x := V])\ U_2[x := V] \to_l U_1'[x := V'][y := U_2'[x := V']]$, а по лемме о подстановке $U_1'[x := V'][y := U_2'[x := V']] =_\alpha U_1'[y := U_2'][x := V']$.

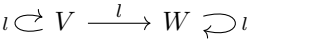
□

Лемма 3. *$\to_l$ обладает свойством ромба $V \to_l V_1, U \to_l V_2 : V_1 \to_l W, V_2 \to_l W$.*

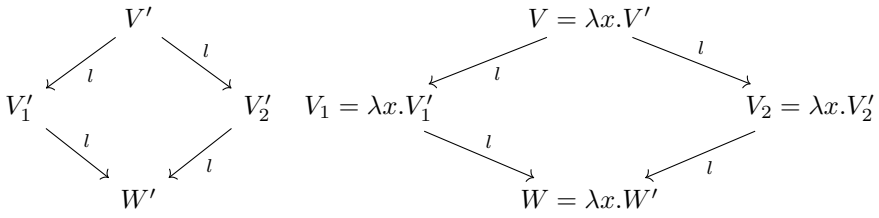


Доказательство. Построим W по индукциии

1. $V = V_1$, тогда в качестве W нужно взять V_2.

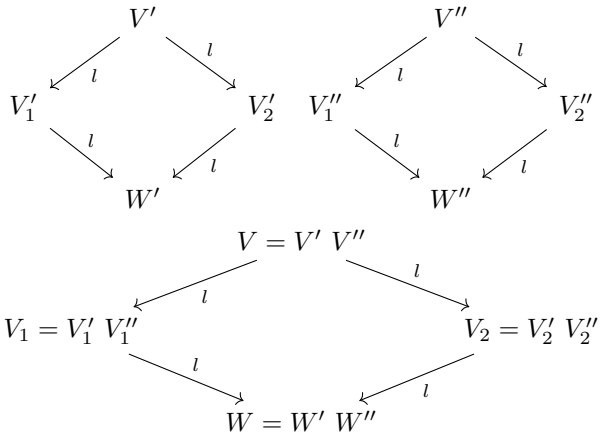


2. $V = \lambda x.V', V_1 = \lambda x.V_1', V' \to_l V_1'$, т.к. $\lambda x.V' \to_l V_2$ значит $V_2 = \lambda x.V_2'$, получается, что $V' \to_l V_1', V' \to_l V_2'$ и по предположению индукции $V_1' \to_l W', V_2' \to_l W'$, а значит $W = \lambda x.W'$.

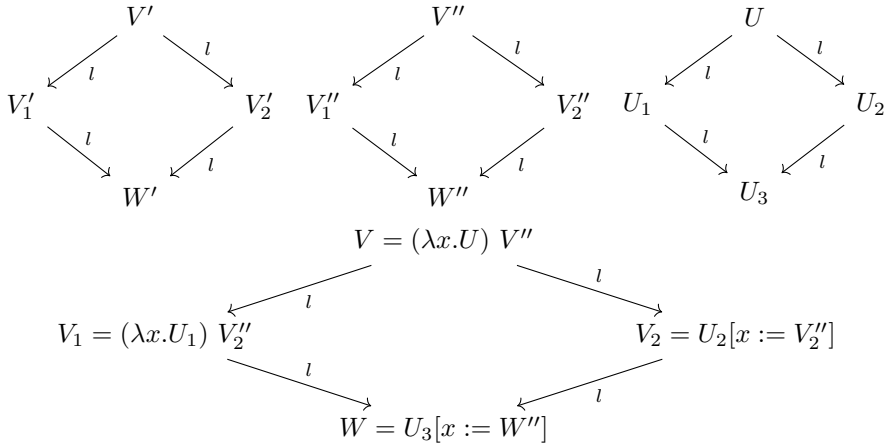


3. $V = V'\ V'', V_1 = V_1'\ V_1''$

   (a) $V_2 = V_2'\ V_2''$



(b) $V' = \lambda x.U$ получается $U \to_l U_2, V'' \to_l V_2'', V = (\lambda x.U)\ V'' \to_l U_2[x := V_2''] = V_2$ и $V_1' = \lambda x.U_1$, тогда в качестве W возьмём $U_3[x := W'']$, где $U_1 \to_l U_3, U_2 \to_l U3$ и $V_1'' \to_l W'', V_2'' \to_l W''$.



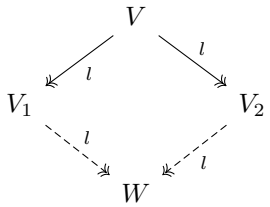
4. $V = (\lambda x.U)\ V'', V_1 = U_1[x := U_1''], V_2 = U_2[x := U_2'']$, тогда $W = U_3[x := W'']$

□

Лемма 4. *$\to_l$ обладает свойством ленты т.е. $U \twoheadrightarrow_l U_n, U \to_l U' : U_n \to_l W, U' \twoheadrightarrow_l W$.*

Доказательство. Доказывается индукцией по n. □

Лемма 5. *$\twoheadrightarrow_l$ обладает свойством ромба.*



Лемма 6. $\twoheadrightarrow_l = \twoheadrightarrow_\beta$

□

2.1.4 Стратегии вычислений

Определение 29. Выражение вида $(\lambda x.T)\ A$ называется редексом.

Определение 30 (Полная бета-редукция)**.** При полной бета-редукции в любой момент может сработать любой редекс.

Определение 31 (Нормальный порядок вычислений)**.** При стратегии нормального порядка вычислений всегда сначала сокращает самый левый, самый внешний редекс.

Определение 32 (Вызов по имени)**.** Стратегия вызова по имени работает так же как при нормальном порядке вычислений, но не позволяет проводить редукцию внутри абстракции.

Определение 33 (Вызов по значению)**.** В соответствие со стратегией вызова по значению сокращаются только самые внешние редексы, и, кроме того, редекс срабатывает только в том случае, если его правая часть уже сведена к нормальной форме(значению), который уже вычислен и не может быть редуцирован.

Пример.

$$I\ (I\ (\lambda z.I\ z))$$

Задача 10. Какие из вышеперечисленных стратегий могут не привести терм к нормальной форме даже если она есть?

Утверждение 2. *Нормальный порядок вычислений всегда приводит терм к нормальной форме, если она есть.*

2.2 Программирование на лямбда исчислении

2.2.1 Логические значения, пары и нумералы Чёрча

Определение 34 (Логические значения)**.**

$$TRUE = \lambda x.\lambda y.x$$

$$FALSE = \lambda x.\lambda y.y$$

Задача 11. Придумать функции NOT, AND, OR, IF.

$$IF = \lambda b.\lambda x.\lambda y.b\ x\ y$$

$$NOT = \lambda b.IF\ b\ FALSE\ TRUE$$

$$NOT = \lambda b.\lambda t.\lambda f.b\ f\ t$$

$$AND = \lambda x.\lambda y.x\ y\ FALSE$$

$$OR = \lambda x.\lambda y.x\ TRUE\ y$$

Определение 35 (Пары)**.** Функция $PAIR$ принимает два значения и запаковывает их в пару так, чтобы к ним можно было обращаться по FST и SND.

$$PAIR = \lambda x.\lambda y.\lambda f.f\ x\ y$$

Задача 12. Придумать функции FST, SND.

$$FST = \lambda p.p\ TRUE$$

$$SND = \lambda p.p\ FALSE$$

Определение 36 (Нумералы Чёрча)**.** Введём на основе лямбда-исчисления аналог натуральных чисел, основанный на идее, что натуральное число — это или ноль, или увеличенное на единицу натуральное число.

$$\bar{0} = \lambda s.\lambda z.z$$

$$\bar{1} = \lambda s.\lambda z.s\ z$$

$$\bar{2} = \lambda s.\lambda z.s\ (s\ z)$$

Задача 13. Придумать термы $SUCC$, $PLUS$, $MULT$, POW, IS_ZERO.

$$SUCC = \lambda n.\lambda s.\lambda z.s\ (n\ s\ z)$$

$$PLUS = \lambda n.\lambda m.\lambda s.\lambda z.n\ s\ (m\ s\ z)$$

$$MULT = \lambda n.\lambda m.\lambda s.\lambda z.n\ (m\ s)\ z$$

$$POW = \lambda n.\lambda m.\lambda s.\lambda z.m\ n\ s\ z$$

$$IS\_ZERO = \lambda n.n\ (\lambda x.FALSE)\ TRUE$$

Задача 14. Придумать функции $PRED$, $MINUS$. Если вы ничего не поняли, не огорчайтесь. Вычитание придумал Клини, когда ему вырывали зуб мудрости. А сейчас наркоз уже не тот.

$$MINUS = \lambda n.\lambda m.m\ PREDn$$

$$PRED = \lambda n.\lambda s.\lambda z.SND\ (n\ (\lambda p.PAIR\ (s\ (FSTp))\ (FSTp))\ (PAIR\ z\ z))$$

Задача 15. Придумать функции для сравнения чисел.

$$EQ = \lambda n.\lambda m.AND\ (IS\_ZERO\ (MINUS\ n\ m))\ (IS\_ZERO\ (MINUS\ m\ n))$$

$$LEQ = \lambda n.\lambda m.(IS\_ZERO\ (MINUS\ n\ m))$$

$$GEQ = \lambda n.\lambda m.(IS\_ZERO\ (MINUS\ m\ n))$$

$$NEQ = \lambda n.\lambda m.NOT\ (AND\ (IS\_ZERO\ (MINUS\ n\ m))\ (IS\_ZERO\ (MINUS\ m\ n)))$$

Определение 37. Список можно представить через его функцию свёртки $FOLD(REDUCE)$. Например, $[x, y, z] = \lambda c.\lambda n.c\ x\ (c\ y\ (c\ z\ n))$.

Задача 16. Как будет выглядеть NIL? Придумайте функцию $CONS$, которая принимает элемент и список, возвращает список вместе с этим элементом. Напишите функцию IS_NIL, $HEAD$ и $TAIL$.

$$NIL = \lambda c.\lambda n.n$$

$$CONS = \lambda h.\lambda t.\lambda c.\lambda n.c\ h\ (t\ c\ n)$$

$$IS\_NIL = \lambda l.l\ (\lambda h.\lambda t.FALSE)\ TRUE$$

$$HEAD = \lambda l.l\ (\lambda h.\lambda t.h)\ NIL = \lambda l.l\ TRUE\ NIL$$

$$TAIL = \lambda l.FST\ (l\ (\lambda x.\lambda p.PAIR\ (SND\ p)\ (CONS\ x\ (SND\ p)))\ (PAIR\ NIL\ NIL))$$

2.2.2 Рекурсия

Теорема 3 (О неподвижной точке)**.** *Для любого λ-терма F существует такой λ-терм X, что $F\ X = X$. Такой терм X называется неподвижной точкой F. Более того, существует такой комбинатор Y, что для любого λ-терма F терм $Y\ F$ является неподвижной точкой F, т.е., $F\ (Y\ F) = Y\ F$. Комбинатор Y называют комбинатором неподвижной точки.*

Доказательство. Пусть $W = \lambda x.F\ (x\ x)$ и $X = W\ W$, тогда:

$$X = W\ W = (\lambda x.F\ (x\ x))\ W = F\ (W\ W) = F\ X$$

□

Задача 17 (Комбинатор Хаскелла Карри)**.** Доказать, что $Y = \lambda f.(\lambda x.f\ (x\ x))(\lambda x.f\ (x\ x))$ комбинатор неподвижной точки.

Задача 18 (Комбинатор Алана Тьюринга)**.** Доказать, что $\Theta = (\lambda x.\lambda y.y\ (x\ x\ y))(\lambda x.\lambda y.y\ (x\ x\ y))$ комбинатор неподвижной точки.

Задача 19. Убедиться, что функция $FACT$ вычисляет факториал числа n.

$$g = \lambda fct.\lambda n.IF\ (IS\_ZERO\ n)\ \bar{1}\ (MULT\ n\ fct\ (PRED\ n))$$

$$FACT = Yg$$

Задача 20. Реализовать функцию вычисляющую n-ое число Фиббоначи.

$$g = \lambda fib.\lambda n.IF\ (IS\_ZERO\ n)\ \bar{0}\ (IF\ (EQ\ n\ \bar{1})\ \bar{1}\ (PLUS\ (fib\ (PRED\ n))\ (fib\ (MINUS\ n\ \bar{2}))))$$

$$FIB = Yg$$