Конспект по теории вычислимости
IV семестр, 2021 год
Современное программирование, факультет математики и компьютерных наук, СПбГУ
(лекции Пузыниной Светланы Александровны)

Тамарин Вячеслав

February 25, 2021

Contents

Исходный код на https://github.com/tamarinvs19/theory_university

Некоторые доказательства были опущены на лекции, но написаны мной. Они выделены оранжевыми символами:

□ Исправляйте, дополняйте, меняйте. Чем меньше недоказанных утверждений, тем лучше! ■

Index

Chapter 1

Вычислимость. Система вычислимости по Клини

1.1 Рекурсивные функции

Лекция 1: †

11 feb

Определение 1

Пусть функция $f\colon \mathbb{N}^k \to \mathbb{N}$, $k \in \mathbb{N}$, где $\mathbb{N} = \{0, 1, 2, \ldots\}$. Такая функция называется k-**местной частичной функцией**. Если $k = 0$, то $f = \text{const}$.

1.1.1 Простейшие функции

Простейшими будем называть следующие функции:

- Нуль местный нуль — функция без аргументов, возвращающая 0;
- Одноместный нуль — $0(x) = 0$;
- Функция следования — $s(x) = x + 1$;
- Функция выбора (проекция) — $I_n^m(x_1, \ldots x_n) = x_m$

1.1.2 Операторы

Определим три оператора:

Определение 2

- Функция f **получается оператором суперпозиции** из функций h и g_i, где

$$h(y_1, \ldots, y_m),\ g_i(x_1, \ldots, x_n);\ 1 \leqslant i \leqslant n,$$

  если

$$f = h(g_1(x_1, \ldots, x_n), \ldots g_m(x_1, \ldots, x_n)).$$

  Оператор обозначается **S**.

- Функция $f^{(n+1)}$[a] **получается оператором примитивной рекурсией** из $g^{(n)}$ и $h^{(n+2)}$, если

$$\begin{cases} f(x_1, \dots x_n, 0) = g(x_1, \dots x_n) \\ f(x_1, \dots x_n, y+1) = h(x_1, \dots x_n, y, f(x_1, \dots x_n, y)) \end{cases}$$

  Оператор обозначается **R**.

- Функция f задается **оператором минимизации** (**M**), если она получается из функции g следующим образом:

$$f(x_1, \dots x_n) = \mu y\big[g(x_1, \dots x_n, y) = 0\big] = \\ = \begin{cases} y & g(x_1, \dots x_n, y) = 0 \wedge g(x_1, \dots x_n, i)^b \neq 0 \ \forall i < y \\ \uparrow^c & else \end{cases}$$

[a] Здесь и далее $f^{(n)}$ обозначается функция, принимающая n аргументов, то есть n-местная
[b] подразумевается, что функция определена в этих точках
[c] не определена

Пример 1.1.1.

$$x - y = \begin{cases} x - y, & x \geqslant y \\ \uparrow, & x < y \end{cases}$$

Можно задать, используя оператор минимизации:

$$x - y = \mu z[|(y + z) - x| = 0].$$

1.1.3 Функции

Определение 3: Примитивно рекурсивная функция

Функция f называется **примитивно рекурсивной** (**ПРФ**), если существует последовательность таких функций $f_1, \dots f_k$, что все f_i либо простейшие, либо получены из предыдущих $f_1, \dots f_{i-1}$ с помощью одного из операторов **S** и **R** и $f = f_k$.

Пример 1.1.2. Докажем, что $f(x, y) = x + y$ — **ПРФ**. По **R** можем получить f так:

$$\begin{cases} f(x, 0) & = x = I_1^1(x) = g \\ f(x, y+1) & = (x + y) + 1 = s(f(x, y)) = s(I_3^3(x, y, f(x, y))) = h \end{cases}$$

Теперь построим последовательность функций f_i, где последним элементом будет f, полученный с помощью **R**:

$$I_1^1,\ s,\ I_3^3,\ h = \mathbf{S}(s, I_3^3),\ f.$$

Определение 4: Частично рекурсивная функция

Функция f называется **частично рекурсивной функцией** (**ЧРФ**), если существует последовательность функций $f_1, \dots f_k$, таких что f_i либо простейшая, либо получается из предыдущих с помощью одного из операторов **S**, **R**, **M**.

Замечание. Частично рекурсивная функция может быть не везде определена. Но примитивно рекурсивная определена везде.

Замечание. Существуют частично рекурсивные функции, которые всюду определены, но при этом не являются **ПРФ**.

Определение 5

Общерекурсивная функция — всюду определенная частично рекурсивная.

Пример 1.1.3. $\mu y[x+y+1=0]$ — нигде не определена, но получается из последовательности других функций с помощью операторов.

Лемма 1. Следующие функции являются **ПРФ**:

1. $\text{const}^{(n)}$
2. $x+y$
3. $x \cdot y$
4. x^y, где 0^0 можем определить, как хотим
5. $\text{sg}(x) = \begin{cases} 0 & x=0 \\ 1 & x \neq 0 \end{cases}$
6. $\overline{\text{sg}}(x) = \begin{cases} 1 & x=0 \\ 0 & x \neq 0 \end{cases}$
7. $x \dot{-} 1 = \begin{cases} u & x=0 \\ x-1 & x>0 \end{cases}$
8. $x \dot{-} y = \begin{cases} 0 & x<y \\ x-y & else \end{cases}$
9. $|x-y|$

□

1. Сначала можем получить нужное число последовательной суперпозицией функции следования (получили константу от одной переменной), затем проецируем I_1^{n+1}, чтобы получить n переменных (первая - наша константа).
2. Доказали выше в примере 1.1.2.
3. $f(x,y) = xy$ определим так:
$$\begin{cases} f(x,0) & = 0 \\ f(x,y+1) & = f(x,y)+x \end{cases}$$
а складывать мы умеем.
4. $f(x,y) = x^y$:
$$\begin{cases} f(x,0) & = 1 = s(0) \\ f(x,y+1) & = f(x,y) * y \end{cases}$$
Умножать тоже можно по третьему пункту.
5. $\text{sg}(x) = \begin{cases} 0 & x=0 \\ 1 & x \neq 0 \end{cases}$
$$\begin{cases} \text{sg}(0) & = 0 \\ \text{sg}(x+1) & = 1 = s(0) \end{cases}$$
6. Аналогично

7. $f(x) = x \dot{-} 1$
$$\begin{cases} f(0) & = 0 \\ f(x+1) & = x = I_1^1(x) \end{cases}$$

8. $f(x,y) = x \dot{-} y$
$$\begin{cases} f(x,0) & = x = I_1^1(x) \\ f(x,y+1) & = f(x,y) \dot{-} 1 \end{cases}$$

9. $f(x,y) = |x-y| = (x \dot{-} y) + (y \dot{-} x)$

Замечание. Обычное вычитание не является **ПРФ**, так как не везде определено на $\mathbb{N}$.

1.1.4 Оператор ограниченной минимизации

Определение 6: Оператор ограниченной минимизации

Функция $f^{(n)}$ задается **оператором ограниченной минимизации** из функций $g^{(n+1)}$ и $h^{(n)}$, если
$$\mu y \leqslant h(\overline{x})[g(\overline{x},y) = 0]^a.$$

Это означает, что
$$f(\overline{x}) = \begin{cases} y & g(\overline{x},y) = 0 \land y \leqslant h(\overline{x}) \land g(\overline{x},i) \neq 0^b \ \forall i < y \\ h(\overline{x}) + 1 & else \end{cases}$$

[a] Здесь и далее $\overline{x} = x_1, \ldots x_n$.
[b] Аналогично, подразумевается, что функция определена в этих точках

Утверждение. Пусть $g^{(n+1)}, h^{(n)}$ — примитивно рекурсивные функции, $f^{(n)}$ получается из g и h с помощью ограниченной минимизации, то f тоже **ПРФ**.

□ Заметим, что f можно получить следующим образом:
$$f(\overline{x}) = \sum_{y=0}^{h(x)} \prod_{i=0}^{y} \mathrm{sg}(g(\overline{x},i)).$$

Внутреннее произведение равно единице только тогда, когда все $g(\overline{x},i) \neq 0$. Если для некоторого y обнуляется $g(\overline{x},y)$, то все произведения, начиная с $y+1$, будут равны нулю, поэтому просуммируются только y единиц. Если же такого y нет, получим сумму из $h(\overline{x})+1$ единицы. Именно это и нужно.

Проверим, что можно получить $a(\overline{x},y) = \sum_{i=0}^{y} g(\overline{x},i)$ и $m(\overline{x},y) = \prod i = 0^y g(\overline{x},i)$ с помощью примитивной рекурсии:
$$\begin{cases} a(\overline{x},0) & = g(\overline{x},0) \\ a(\overline{x},y+1) & = a(\overline{x},y) + g(\overline{x},y+1) \end{cases} \quad \begin{cases} m(\overline{x},0) & = g(\overline{x},0) \\ m(\overline{x},y+1) & = m(\overline{x},y) \cdot g(\overline{x},y+1) \end{cases}$$

■

Замечание. $0(x)$ можно исключить из определения простейших функций, так как она получается с помощью **R** для нульмерного 0 и $I_2^2(x,y)$:
$$0(y) = \begin{cases} 0(0) & = 0 \\ 0(y+1) & = I_2^2(y,0) \end{cases}$$

1.1.5 Предикаты

Определение 7

Предикат — условие задающее подмножество: $R \subset \mathbb{N}^k$.

Предикат называется **примитивно рекурсивным (общерекурсивным)**, его характеристическая функция примитивно рекурсивная (общерекурсивная).

$$\chi_R(\overline{x}) = \begin{cases} 1, & \overline{x} \in R \\ 0, & \overline{x} \notin R \end{cases}$$

Утверждение.

- Если R, Q — примитивно рекурсивные (общерекурсивные) предикаты, то $P \vee Q$, $P \wedge Q$, $P \to Q$, $\neg P$ тоже примитивно рекурсивные (общерекурсивные).
- Предикаты $=$, $\leqslant$, $\geqslant$, $<$, $>$ тоже примитивно и общерекурсивны.

□

- Проверим, что характеристические функции примитивно / общерекурсивны:

$$\begin{aligned} \chi_{P \wedge Q}(\overline{x}) &= \chi_P(\overline{x}) \cdot \chi_Q(\overline{x}) \\ \chi_{P \vee Q}(\overline{x}) &= \operatorname{sg}(\chi_P(\overline{x}) + \chi_Q(\overline{x})) \\ \chi_{P \to Q}(\overline{x}) &= \overline{\operatorname{sg}}(\chi_P(\overline{x}) + \overline{\operatorname{sg}}(\chi_Q(\overline{x}))) \\ \chi_{\neg P}(\overline{x}) &= \overline{\operatorname{sg}}(\chi_P(\overline{x})) \end{aligned}$$

- Аналогично выразим, через простейшие:

$$\chi_{x=y}(x) = \overline{\operatorname{sg}}(|x - y|) = \begin{cases} 1, & x = y \\ 0, & x \neq y \end{cases}$$
$$\chi_{x<y}(x) = \operatorname{sg}(x \dot{-} y)$$

  Остальные можем выразить также или через уже проверенные $<$ и $\neg$.

■

Лемма 2. Следующие функции являются примитивно рекурсивными:

1. $\left\lfloor \frac{x}{y} \right\rfloor$, считаем, что $\left\lfloor \frac{x}{0} \right\rfloor = x$
2. $\operatorname{Div}(x, y) = \begin{cases} 1, & y \mid x \\ 0, & else \end{cases}$
3. $\operatorname{Prime}(x) = \begin{cases} 1, & x \in \mathbb{P} \\ 0, & else \end{cases}$
4. $f(x) = p_x$, где p_x — x-тое простое число, $p_0 := 2$
5. $\operatorname{ex}(i, x)$ — степень простого числа p_i разложении x, $\operatorname{ex}(i, 0) := 0$

□

1. $f(x,y) = \left\lfloor \frac{x}{y} \right\rfloor$. Найдем минимальное k, что $f'(x,y,k) = yk > x$. Чтобы получить $f(x,y) = \min(k \mid f'(x,y,k)) - 1$. Используем оператор минимизации:

$$f(x,y) = \mu k[\neg f'(x,y,k) = 0] - 1.$$

2. $\mathrm{Div}(x,y) = \left\lfloor \frac{x}{y} \right\rfloor \cdot y = x$

3. Определим $\mathrm{Div}'(x,y) = (y \leqslant 1) \vee (\neg \mathrm{Div}(x,y))$, эта функция проверяет, что y не является нетривиальным делителем x.

   Теперь, используя ограниченную минимизацию, выразим $\mathrm{Prime}(x)$:

$$\mathrm{Prime}(x) = \left(\mu y \leqslant h(x)[\mathrm{Div}'(x,y) = 0]\right) = x, \text{ где } h(x) = x - 1.$$

   То есть мы посмотрели на все меньшие числа, если среди них найдется нетривиальный делитель, то число не простое.

4. Пусть $f'(x) =$ количество простых $\leqslant x$.

$$\begin{cases} f'(0) & = 0 \\ f'(x+1) & = \mathrm{Prime}(x+1) + f(x) \end{cases}$$

   Теперь можно вычислить $f(x)$: для этого определим функцию $g(x,y) = (f'(y) = x)$,

$$f(x) = \mu y[\neg f'(x,y) = 0].$$

5. Чтобы найти степень вхождения простого числа p_i в x, сначала находим это простое число по номеру, затем находим минимальное k, что x не делится на p_i^k и вычитаем единицу.

■

1.1.6 Теоремы про рекурсии

Теорема 1.1.1 (Канторовская нумерация). Пусть $\pi\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$:

$$\pi(x,y) = \frac{1}{2}(x+y)(x+y+1) + y.$$

- Тогда для любого z существует единственное представление $z = \pi(x,y)$.
- Причем функции $x(z), y(z)$ примитивно рекурсивные.

□

- Запишем $\pi(x,y) = \binom{x+y+1}{2} + y$. Заметим, что для $n > m$ верно

$$\binom{n}{2} - \binom{m}{2} \geqslant \binom{n}{2} - \binom{n-1}{2} = n - 1.$$

  Предположим, что $x + y > x' + y'$ и $\pi(x,y) = \pi(x',y')$. Тогда

$$y' - y = \binom{x+y+1}{2} - \binom{x'+y'+1}{2} \geqslant x + y > x' + y'.$$

Но $y \geqslant 0$, $x' \geqslant 0$, поэтому $y' - y \leqslant y$, а $x' + y' \geqslant y'$, а тогда $y' - y \leqslant x' + y'$, что противоречит полученному выше неравенству.

Если $x + y = x' + y'$, то $0 = \pi(x, y) - \pi(x', y') = y - y'$. Тогда $y = y' = 0$, поэтому $x = x'$.

Доказали, что из равенства $\pi(x, y) = \pi(x', y')$ следует равенство $(x, y) = (x', y')$.

- Можно по-честному все посчитать и выразить $x(z)$, $y(z)$. Пусть

$$\begin{aligned} w &= x + y \\ t &= \frac{1}{2}w(w+1) = \frac{w^2 + w}{2} \\ z &= t + y \end{aligned}$$

Решим квадратное уравнение, чтобы выразить w через t (отрицательный корень можем сразу отбросить):

$$w = \frac{-1 + \sqrt{8t+1}}{2}.$$

Запишем неравенство:

$$t \leqslant z = t + y < t + (w+1) = \frac{(w+1)^2 + (w+1)}{2}.$$

Аналогично выразим $w + 1$ через z: имеем $z < \frac{(w+1)^2+(w+1)}{2}$, решаем неравенство, а далее вспоминаем, что все числа положительные и можно забыть про отрицательные корни. Отсюда

$$w \leqslant \frac{-1 + \sqrt{8z+1}}{2} < w + 1.$$

Тогда

$$\begin{aligned} w &= \left\lfloor \frac{-1 + \sqrt{8z+1}}{2} \right\rfloor \\ t &= \frac{w^2 + w}{2} \\ y &= z - t \\ x &= w - y \end{aligned}$$

Таким образом, мы выразили через z обе координаты. Единственный момент — нужно извлекать корень, в натуральную степень возводить мы умеем, поэтому можем с помощью ограниченной минимизации перебрать все меньшие числа, возвести их в квадрат и сравнить с нашим числом.

■

Теорема 1.1.2 (Возвратная рекурсия)**.** Зафиксируем s. Пусть

$$\begin{cases} f(\overline{x}, 0) & = g(\overline{x}) \\ f(\overline{x}, y+1) & = h(\overline{x}, y, f(\overline{x}, t_1(y)), \dots f(\overline{x}, t_s(y))) \end{cases}$$

где $\forall 1 \leqslant i \leqslant s\ t_i(y) \leqslant y, g^{(n)}, h^{(n+1+s)}, t_i^{(1)},$.

Тогда, если g, h, t_i — примитивно / общерекурсивные, то и f тоже.

Основная идея этой теоремы — можем использовать все ранее вычисленные значения функции, а не только предыдущее.

□ Построим с помощью примитивной рекурсии функцию $m(\overline{x}, y)$, которая возвращает закодированную последовательность $f(\overline{x}, i)$, $0 \leqslant i \leqslant y$.

Кодировать будем так: каждому $f(\overline{x}, i)$ будет соответствовать p_i (i-ое простое число) в степени $1 + f(\overline{x}, i)$.

Если мы построим эту функцию, то $f(\overline{x}, y)$ — уменьшенная на 1 степень y-ого простого, обозначим функцию, которая это делает:

$$f(\overline{x}, y) = \mathrm{ith}(y, m(\overline{x}, y)).$$

Вернемся к построению m:

$$\begin{cases} m(\overline{x}, 0) & = 2^{1+g(\overline{x})} \\ m(\overline{x}, y+1) & = m(\overline{x}, y) \cdot p_{y+1}^{1+h(\overline{x}, y, \mathrm{ith}(t_1(y), m(\overline{x}, y)), \ldots \mathrm{ith}(t_k(y), m(\overline{x}, y)))} \end{cases}$$

Теорема 1.1.3 (Совместная рекурсия)**.** Пусть $f_i^{(n+1)}$, $1 \leqslant i \leqslant k$,

$$\begin{cases} f_i(\overline{x}, 0) & = g_i(\overline{x}) \\ f_i(\overline{x}, y+1) & = h_i(\overline{x}, y, f_1(\overline{x}, y), \ldots f_k(\overline{x}, y)) \end{cases}$$

Если $g_i^{(n)}, h_i^{(k+2)}$, $1 \leqslant i \leqslant k$ — примитивно / общерекурсивные, то f_i тоже.

Основная идея этой теоремы — можем использовать y-е значение каждой из k функций.

Заметим, что канторовскую функцию можно, последовательно применив несколько раз, расширить до k-местной. Обозначим полученную функцию за c, а обратные за $c_1, \ldots c_k$.

Давайте просто объединим все f_i в одну функцию

$$m(\overline{x}, y) = c\left(f_1(\overline{x}, y), \ldots f_k(\overline{x}, y)\right).$$

Теперь каждую f_i можно вычислить

$$f_i(\overline{x}, y) = c_i(m(\overline{x}, y)).$$

Чтобы получить m достаточно использовать примитивную рекурсию:

$$\begin{cases} m(\overline{x}, 0) & = c\left(g_1(\overline{x}), \ldots g_k(\overline{x})\right) \\ m(\overline{x}, y+1) & = c(\\ & \quad h_1\left(\overline{x}, y, c_1(m(\overline{x}, y)), \ldots c_k(m(\overline{x}, y))\right), \\ & \quad \quad \vdots \\ & \quad h_k\left(\overline{x}, y, c_1(m(\overline{x}, y)), \ldots c_k(m(\overline{x}, y))\right) \\ &) \end{cases}$$

Теорема 1.1.4 (Кусочное задание функции)**.** Пусть $R_0, \ldots R_k$ — отношения[a], такие что $\bigsqcup_{i=0}^{k} R_i = \mathbb{N}^m$[b].

Для $|\overline{x}| = n$ кусочно зададим функцию $f^{(n)}$:

$$f(\overline{x}) = \begin{cases} f_0(\overline{x}), & \text{если } R_0(\overline{x}) \\ f_1(\overline{x}), & \text{если } R_1(\overline{x}) \\ \quad \vdots & \quad \vdots \\ f_k(\overline{x}), & \text{если } R_k(\overline{x}) \end{cases}$$

Если $f_i^{(n)}, R_i$ — примитивно / общерекурсивны, то и f тоже.

[a]Набор предикатов
[b]То есть для $i \neq j$ верно $R_i \cap R_j = \varnothing$.

□ Рассмотрим характеристические функции χ_{R_i} для R_i. Тогда

$$f(\overline{x}) = \sum_{i=0}^{k} f_i(\overline{x}) \cdot \chi_{R_i}(\overline{x}).$$

А это просто сумма произведений, которые мы можем вычислять. ■

1.2 Равносильность МТ и ЧРФ

Теорема 1.2.1. Функция вычисляется машиной Тьюринга тогда и только тогда, когда она частично рекурсивная (то есть вычислима по Клини).

□

$\boxed{2 \Longrightarrow 1}$ Если $f(x_1, \dots x_n) = y$, то считаем, что МТ получаем $1^{x_1}01^{x_2}0\dots01^{x_n}$ и должна выдать 1^y; если f не определена, МТ должна зацикливаться и наоборот.

- Для простых функций можем построить МТ напрямую:
  - Если мы хотим выдавать нуль, просто стираем вход.
  - Если нужно увеличить число на один, приписываем 1 в конец справа.
  - Если нужно вернуть k-ую проекцию, стираем все до начала k-ого числа (то есть нужно отсчитать $k-1$ нуль на входе), далее стереть все после.
- Для операторов **S**, **R**, **M**:

  S: Пусть есть набор функций $h^{(n)}, g_1^{(m)}, \dots, g_n^{(m)} \longrightarrow f^{(m)}$, для каждой из которых есть машина Тьюринга M_h и M_{g_i}.
  Хотим построить МТ M_S для S.
  Сделаем это так:
  - Копируем весь вход n раз:
  $$\left(1^{x_1}01^{x_2}\dots01^{x_n}*\right)^n.$$
  - Запускаем M_{g_i} на соответствующей части полученного входа.
  Если нужно что-то записать, то будем сдвигать всю правую часть на нужное число клеток, чтобы освободить для место.
  МТ запускаем псведопараллельно (по очереди даем поработать).
  В каждой часть после окончания работы оставляем только ответ:
  $$1^{y_1} * 1^{y_2} \dots * 1^{y_n},$$
  где $y_i = g_i(x_1, \dots x_m)$.
  - Запускаем на этом результате M_h.

Лекция 2: †

18 feb

R: Пусть рекурсия задает $f^{(m+1)}(x_1, \dots x_m, y)$ из $g^{(m)}$ и $h^{(m+2)}$.

$$\begin{cases} f(\overline{x}, 0) & = g(\overline{x}) \\ f(\overline{x}, y+1) & = h(\overline{x}, y, f(\overline{x}, y)) \end{cases}$$

Считаем, что для g, h уже есть МТ (M_g и M_h), и мы хотим построить M_f, которая будет вычислять f.

Построим вспомогательные МТ:

- M_1: для входа $1^{x_1}0\dots01^{x_m}01^y$ построим $1^y01^{x_1}0\dots01^{x_m}001^{g(x_1,\dots x_m)}$. Для этого просто запустим M_g на входе, но не будем стирать его, а результат просто припишем после двух нулей справа.
- M_2: для входа $1^y01^{x_1}0\dots01^{x_m}01^u01^z$ построим $1^y01^{x_1}0\dots01^{x_m}01^{u+1}01^{h(x_1,\dots x_m,u,z)}$, аналогично, используя M_h, допишем в конец вместо z результат h и допишем единицу к 1^u. Здесь $u+1$ обозначает текущее значение y', а значение h — значение $f(y')$.
- M_3: для входа $1^y01^{x_1}0\dots01^{x_m}01^u0^z$ оставим только 1^z.
- Φ: для входа $1^y01^{x_1}0\dots01^{x_m}01^u01^z$ проверим, что $u \neq y$.

Теперь соберем все вместе: сначала запустим M_1, далее пока Φ возвращает неравенство, запускаем M_2 (увеличиваем u на один, вычисляем следующее значение функции), и в конце стираем лишнее, запустив M_3.

M: Хотим по МТ M_g построить M_f, вычисляющую

$$f(\overline{x}, y) = \begin{cases} y & g(\overline{x}, y) = 0 \wedge g(\overline{x}, z) \neq 0 \quad \forall z < y \\ \uparrow & else \end{cases}$$

Аналогично построим несколько вспомогательных МТ:

- N_1 : приписывает 0 ко входу:

$$1^{x_1}0\dots01^{x_m} \longrightarrow 1^{x_1}0\dots01^{x_m}0.$$

- N_2 : дублирует вход, разделяя решеткой: $w \longrightarrow w\#w$
- N_3 : в продублированному входе меняет вторую половину на результат M_g

$$1^{x_1}0\dots01^{x_m}01^y\#1^{x_1}0\dots01^{x_m}01^y \xrightarrow{M_g} 1^{x_1}0\dots01^{x_m}01^y\#1^{g(x_1,\dots x_m,y}.$$

- N_4 : очищает все после решетки и дописывает единицу в конец

$$1^{x_1}0\dots01^{x_m}01^y\#w \longrightarrow 1^{x_1}0\dots01^{x_m}01^{y+1}.$$

- N_5 : стирает все, кроме ответа

$$1^{x_1}0\dots01^{x_m}01^y\#w \longrightarrow 1^y.$$

- Φ : проверяет, что после решетки что-то еще есть $w\#v \longrightarrow v \neq \varepsilon$.

Теперь можем построить M_μ так:

$$N_1;\ N_2;\ N_3; \text{while } \Phi \text{ do } N_4,\ N_2,\ N_3;\ N_5.$$

$\boxed{1 \Longrightarrow 2}$ Теперь мы хотим промоделировать работу МТ с помощью частично рекурсивной функции. На вход должны либо выдать результат, либо зациклится. Так как машины Тьюринга работают со строками, а функции с натуральными числами, нужно придумать правила кодирования.

Пусть есть конфигурация МТ

$$\alpha q_i a_j \beta,$$

где α — строка слева от головки, q_i — состояние, a_j — текущий символ, β — справа от головки.

Пронумеруем рабочий алфавит $\Gamma = \{a_0, \dots a_{m-1}\}$, где a_0 — пустой символ (␣).

Кодирование конфигураций Теперь можем конфигурацию записать как

$$\tilde{\alpha}, \tilde{q_i}, \widetilde{a_j}, \tilde{\beta},$$

где $\tilde{\alpha}$ — число, соответствующее α в m-ичной записи, $\tilde{q_i}$ — просто номер состояния, $\widetilde{a_j}$ — номер в алфавите (j), $\tilde{\beta}$ — число, соответствующее β в m-ичной записи, записанное справа налево.

Сдвиги обозначать будем d: вправо $d = 1$, влево $d = 2$.

Терминальное состояние — z. Множество состояний тоже пронумеруем и получим множество состояний $\widetilde{Q} = \{0, 1, \dots |Q| - 1\}$.

Пример 1.2.1. Рассмотрим небольшой пример. $\Gamma = \{a_0, a_1\}$, тогда состояние $\underbrace{a_1a_0a_1a_1a_0}_{\alpha} q_3 \underbrace{a_1a_1a_0a_1a_1}_{\beta}$ будет записано так: $(22, 3, 1, 13)$.

Кодирование команд Пусть есть переход $(q, a) \to (p, b, d)$. Сопоставим p, b, d тройку функций $\varphi_q, \varphi_a, \varphi_d$:

$$\begin{aligned}
\varphi_a &\colon \widetilde{Q} \times \tilde{\Gamma} \to \tilde{\Gamma} \\
\varphi_q &\colon \widetilde{Q} \times \tilde{\Gamma} \to \widetilde{Q} \\
\varphi_d &\colon \widetilde{Q} \times \tilde{\Gamma} \to \{1, 2\}
\end{aligned}$$

Эти функции будут примитивно рекурсивными, так как заданы на конечном множестве, на остальных можем доопределить нулем.

Преобразование конфигураций Пусть у нас есть переход между двумя конфигурациями:

$$K = \alpha q_i a_j \beta \to \alpha' q_i' a_j' b' = K'.$$

Зададим функцию на числах, которая проделает этот переход $\Phi\colon K \to K'$. На самом деле эта функция состоит из четырех, которые мы сейчас и определим.

Пусть

$$\begin{aligned}
\tilde{q_i}'(\tilde{\alpha}, \tilde{q_i}, \widetilde{a_j}, \tilde{\beta}) &= \varphi_q(\tilde{q_i}, \tilde{a_i}) \\
\tilde{\alpha}'(\tilde{\alpha}, \tilde{q_i}, \widetilde{a_j}, \tilde{\beta}) &= \begin{cases} \tilde{\alpha} \cdot m + \varphi_a(\tilde{q_i}, a_j), & \varphi_d(\tilde{q_i}, \widetilde{a_j}) = 1 \\ \left\lfloor \frac{\tilde{\alpha}}{m} \right\rfloor, & \varphi_d(\tilde{q_i}, \widetilde{a_j}) = 2 \end{cases} \\
\tilde{\beta}'(\tilde{\alpha}, \tilde{q_i}, \widetilde{a_j}, \tilde{\beta}) &= \begin{cases} \tilde{\beta} \cdot m + \varphi_a(\tilde{q_i}, a_j), & \varphi_d(\tilde{q_i}, \widetilde{a_j}) = 1 \\ \left\lfloor \frac{\tilde{\beta}}{m} \right\rfloor, & \varphi_d(\tilde{q_i}, \widetilde{a_j}) = 2 \end{cases} \\
\widetilde{a_j}' &= \begin{cases} \tilde{\beta} \mod m, & \varphi_d(\tilde{q_i}, \widetilde{a_j}) = 1 \\ \tilde{\alpha} \mod m, & \varphi_d(\tilde{q_i}, \widetilde{a_j}) = 2 \end{cases}
\end{aligned}$$

Заметим, что все эти формулы примитивно рекурсивные[1].

Общая работа МТ Пусть $K(0) = \left(\widetilde{\alpha_0}, \widetilde{q_0}, \widetilde{a_0}, \widetilde{\beta_0}\right)$ — начальная конфигурация. Чтобы получить новую конфигурацию для шага t, посчитаем все четыре параметра:

$$\begin{aligned}
&K(t) = (\\
&\quad K_\alpha(\widetilde{\alpha_0}, \widetilde{q_0}, \widetilde{a_0}, \widetilde{\beta_0}, t) \\
&\quad K_q(\widetilde{\alpha_0}, \widetilde{q_0}, \widetilde{a_0}, \widetilde{\beta_0}, t) \\
&\quad K_a(\widetilde{\alpha_0}, \widetilde{q_0}, \widetilde{a_0}, \widetilde{\beta_0}, t) \\
&\quad K_\beta(\widetilde{\alpha_0}, \widetilde{q_0}, \widetilde{a_0}, \widetilde{\beta_0}, t) \\
&\quad)
\end{aligned}$$

[1]Единственное, чего нет явно в лемме 1 выше, это остаток по модулю, но его легко получить из деления нацело.

Теперь запишем совместную рекурсию для $K_\alpha, K_q, K_a, K_\beta$:

$$\begin{cases} K_\alpha(\widetilde{\alpha_0}, \widetilde{q_0}, \widetilde{a_0}, \widetilde{\beta_0}, 0) & = \widetilde{\alpha_0} \\ K_\alpha(\widetilde{\alpha_0}, \widetilde{q_0}, \widetilde{a_0}, \widetilde{\beta_0}, t+1) & = \widetilde{\alpha}'\Big(K_\alpha(\dots,t), K_q(\dots,t), K_a(\dots,t), K_\beta(\dots,t)\Big) \end{cases}$$

Для остальных точно также.

Результат Пусть начальное состояние $q_0 a_0 \beta_0$ (стоим на самом левом символе), конечное — $q_z a_z \beta_z$, причем z встречаем впервые. То есть нам нужно вычислить функцию, которая переводит $\widetilde{a_0} + \widetilde{b_0} m \longrightarrow a_z + \beta_z m$, если машина Тьюринга пришла сюда и не определена, если МТ зацикливается:

$$t_z = \mu t[K_q(t) = z].$$

Тогда результатом работы МТ будет

$$\begin{aligned} \varphi(x) = & m \cdot K_\beta\Big(0, 0, x \mod m, \lfloor \tfrac{x}{m} \rfloor, \\ & \mu t[K_q(0, 0, x \mod m, \lfloor \tfrac{x}{m} \rfloor, t) = z]\Big) + \\ & + K_a(0, 0, x \mod m, \lfloor \tfrac{x}{m} \rfloor) \end{aligned}$$

■

Следствие 1. Любую частично рекурсивную функцию можно представить так, что минимизация использовалась только один раз.

□ Сначала запишем для нее МТ, а потом постоим обратно функцию. В итоге получим эквивалентную функцию, причем по построению оператор минимизации использовался лишь один раз. ■

Следствие 2. Функция вычислимая за примитивно рекурсивное время (время, ограниченное примитивно рекурсивной функцией), является примитивно рекурсивной.

□ В построении функции использовали минимизацию по числу шагов МТ, поэтому, если работаем примитивно рекурсивное время, можем применить ограниченную минимизацию. ■

1.2.1 Функция Аккермана

Можно построить общерекурсивную функцию, которая растет быстрее любой примитивно рекурсивной. Из этого следует, что **ПРФ** не совпадает с **ОРФ**.

Определение 8: Функция Аккермана

Функция Аккермана — функция от двух аргументов $\alpha_n(x)$, которая определяется следующим образом:

$$\begin{cases} a_0(x) & = x + 1 \\ a_{n+1}(x) & = a_n^{[x+2]}(x) = \underbrace{a_n(a_n(\dots(x)))}_{x+2 \text{ раза}} \end{cases}$$

Теорема 1.2.2. $\alpha_n(n) \colon \mathbb{N} \to \mathbb{N}$ растет быстрее любой примитивно рекурсивной.

□ «Доказательство – упражнение», занимает пару страниц, в ближайшее время появится здесь. ■

Chapter 2

Разрешимые и перечислимые множества

2.1 Определения

Определение 9: Разрешимое множество

Множество $X \subseteq \mathbb{N}^k$ называется **разрешимым**, если его характеристическая функция вычислима[a].

[a] Это может быть частично рекурсивная функция, машина Тьюринга, λ-функция...

Замечание. Любое конечное множество разрешимо. Пересечение, объединение, разность разрешимых тоже разрешимо.

Теорема 2.1.1. Множество $X \subseteq \mathbb{N}$ разрешимо тогда и только тогда, когда X — множество значений всюду определенной вычислимой неубывающей функции (или пустое множество).

□

$\boxed{1 \Longrightarrow 2}$ Можем в характеристической функции $\chi_X(n)$ возвращать n вместо 1, а в остальных значениях прошлое выданное. Эта функция подходит под описание.

$\boxed{2 \Longrightarrow 1}$ Пусть есть функция f. Из нее хотим построить χ_X. Посчитаем $\chi_X(n)$ так:

- вычислим $f(0)$;
- если значение больше n, то в следующих входах, значения будут еще больше, поэтому можем сразу вернуть 0;
- если меньше, то посчитаем $f(1)$ и сравним еще раз;
- так как функция неубывающая, мы либо найдем значение больше n (тогда вернем 0), либо равное (тогда вернем единицу).

■

Определение 10: Перечислимое множество

Множество $X \subseteq \mathbb{N}^k$ называется **перечислимым**, если его *полухарактеристическая* функция вычислима:

$$\chi_X(n) = \begin{cases} 1, & n \in X \\ \uparrow, & n \notin X \end{cases}$$

2.2 Перечислимые множества

Определение 11

Подмножество $\mathbb{N}^k$ называется **перечислимым**, если существует алгоритм, который выводит все его элементы в некотором порядке.

Теорема 2.2.1 (Об эквивалентных определений).

1. область определения вычислимой функции
2. область значений вычислимой функции
3. его полухарактеричтическая функция вычислима
4. область значений всюду определена вычислимой функцией

▢

$\boxed{\text{перечислимость} \implies 2}$ Чтобы посчитать $\chi_X(n)$, запускаем алгоритм, перечисляющий элементы множества, ждем n.

Если вывелось n, то выводим $\chi_X(n) = 1$, а иначе мы зациклились, то есть получили расходимость.

$\boxed{3 \implies 1}$ Очевидно — для χ_X

$\boxed{1 \implies \text{перечислимость}}$ Пусть область определения вычисляется алгоритмом B.

Будем запускать B по шагам:

- 1 шаг на 0
- 2 шага на 0, 2 шага на 1
- 3 шага на $0, 1, 2$
- и так далее
- как только B закончил работу на некотором элементе, выводим этот элемент.

Этот алгоритм A перечисляет наше множество.

$\boxed{2 \implies \text{перечислимость}}$ Аналогично, но выводим значение функции на элементе, на котором мы останавливаемся.

$\boxed{1 \implies 2}$ Пусть X — область определения функции, которая вычисляется алгоритмом A. Рассмотрим следующую функцию:

$$b(n) = \begin{cases} n, & \text{если } A \text{ заканчивает работать на } n \\ \uparrow, & \text{если } A \text{ зацикливается на } n \end{cases}$$

Теперь X — область значений $b(n)$, а она вычислима.

$\boxed{\text{перечислимость} \implies 4}$ Пусть A — алгоритм, перечисляющий X. Рассмотрим любой $n_0 \in X$.

Построим функцию f, которая всюду определена и X — ее область значений.

$$f(n) = \begin{cases} t, & \text{если на } n\text{-ом шаге работы } A \text{ появляется } t \\ n_0, & \text{если ничего не появляется} \end{cases}$$

$\boxed{4 \implies 2}$ Очевидно

■

Замечание. Все области значений и определений не применимы к пустому множеству, которое тоже перечислимое.

Теорема 2.2.2. Объединение и пересечение перечислимых множеств тоже перечислимое.

Теорема 2.2.3 (Пост)**.** A разрешимо тогда и только тогда, когда A и $\overline{A}$ перечислимые.

□

$\boxed{1 \implies 2}$ A разрешимо, можем рассмотреть $\chi_A(n)$, которая вычислима.

Построим полухрарактеристические для A и $\overline{A}$:

- если $n \in A$, то $\chi'_A = 1$, иначе $\chi'_A(n)$ расходится
- для $\overline{A}$ аналогично, только наоборот.

$\boxed{2 \implies 1}$ Запускаем одновременно по шагам алгоритм, перечисляющий A и $\overline{A}$, ждем появления n-ого. Рано или поздно должно появиться. Теперь, если его выдал алгоритм для A, то $\chi_A(n) = 1$, а если для $\overline{A}$, то $\chi_A(n) = 0$.

■

Определение 12: Проекция

Подмножество $P \subseteq \mathbb{N}$ называется **проекцией** $Q \subseteq \mathbb{N}^2$, если

$$x \in P \iff \exists y\colon (x, y) \in Q.$$

Теорема 2.2.4 (О проекции)**.** P перечислимое тогда и только тогда, когда P — проекция некоторого разрешимого Q.

□

$\boxed{1 \implies 2}$ Пусть A — алгоритм, перечисляющий P. Тогда

$$Q \coloneqq \{(n, t) \mid n \text{ появляется в течение } t \text{ шагов работы } A\}.$$

$\boxed{2 \implies 1}$ Проекция перечислимого перечислима: берем алгоритм, которые перечисляет Q, но оставляем только первую координату.

■

Теорема 2.2.5 (О графике)**.** Частичная функция $f\colon \mathbb{N} \to \mathbb{N}$ вычислима тогда и только тогда, когда ее график

$$F = \{(x, y) \mid f(x) \text{ определена и } f(x) = y\}$$

перечислим.

□ Упражнение ■

Замечание. Различные обозначения в других источниках

перечислимое	разрешимое
вычислимо перечислимое	вычислимое
полурекурсивное	рекурсивное
полуразрешимое	decidable
enumerable	recursive
semi-decidable	...

2.3 Универсальные функции

Рассмотрим функцию $U^{(m+1)}(n, \overline{x})$.

Обозначим за $U_n(\overline{x})$ — функция от m аргументов (получается из U фиксацией первого аргуменаа). Обозначается проекцией ???

Определение 13: универсальная функция

$U(n, \overline{x})$ — универсальная для класса K функция от m аргументов, если

- $\forall n\colon U_n(\overline{x}) \in K$
- $\forall f \in K\ \exists n\colon f = U_n$

Замечание. Универсальная функция существует только для счетных K

Замечание. Все рассматриваемые функции частичные

Теорема 2.3.1. Существует вычислимая функция 2^x аргументов, универсальная для класса вычислимых функций 1-ого аргумента.

□ Запишем все коды МТ, вычисляющих функции 1-ого аргумента, в порядке возрастания (сначала по длинне, затем в алфавитном).

Пусть $U(i, x)$ — результат работы МТ M_i на входе x. Сечение U_i соответствует МТ M_i. ■

Обозначение. *$\mathcal{F}^m$ — вычислимые функции от m аргументов $\mathcal{F}^m_*$ — всюду определенные вычислимые функции от m аргументов*

Следствие 3. Существует $U \in \mathcal{F}^{m+1}$, универсальная для $\mathcal{F}^m$.

□ Используем канторовскую нумерацию m-ок $c(\overline{x})$. Универсальной для 2^x будет

$$U'(n, \overline{x}) \coloneqq U(n, c(\overline{x})).$$

Проверим это. Рассмотрим произвольную функцию $f(\overline{x}) \in \mathcal{F}^m$.

Определим

$$g(\overline{x}) \coloneqq f(c_1(x), \ldots c_m(x)).$$

g вычислима, U — универсальна, поэтому

$$\exists n\colon U_n(x) = g(x).$$

$$U'(n, \overline{x}) \overset{\text{по определению } U}{=} U(n, c(\overline{x})) \overset{\text{по определению } g}{=} g(c(\overline{x})) \overset{\text{по определению } g}{=} f(\overline{x}) = f(c_1(c(\overline{x})), \ldots c_m(c(\overline{x}))).$$

То есть U' действительно универсальная. ■

Теорема 2.3.2. Не существует $V \in \mathcal{F}^2_*$ универсальной для F^*.

□ Предположим, что такая функция $U \in \mathcal{F}^2_*$ существует.

Рассмотрим диагональную функцию:

$$d'(n) = U(n, n) + 1 \in \mathcal{F}_*.$$

$d'(n)$ отличается от всех сечений U: если $\forall x\colon U(n, x) = d'(x)$, подставим $x = n\colon U(n, n) \neq U(n, n) + 1$. Противоречие. ■

Замечание. Для класса частичных функций такое рассуждение не проходит, так как они могут быть не определены и прибавление единицы ничего не меняет для неопределенности.

Теорема 2.3.3. Существует вычислимая частичная функция, не имеющая всюду определенного вычислимого продолжения[a].

[a]то есть нельзя доопределить до всюду определенной вычислимой

☐ Подходит функция $d^{'}(n) = U(n,n) + 1$, где U — универсальная вычислимая функция[1].

Пусть ее можно доопределить до вычислимой $d^{''}$:

- при n, где $d(n,n)$ определена, $d^{''}$ на один больше
- а где не определена, $d^{''}$ определена.

Поэтому $d^{''}$ отличается от всех сечений универсальной функции. Противоречие. ■

Аналогично можно ввести определения для перечислимых множеств $W \subset \mathbb{N}^2$

2.3.1 Перечислимое неразрешимое множество

Теорема 2.3.4. Существует перечислимое неразрешимое множество.

☐ Рассмотрим вычислимую $f(x)$, не имеющую всюду определенного вычислимого продолжения. Ее область определения F — перечислимое и неразрешимое.

- F перечислимое, так как оно является областью определения вычислимой функции
- F не разрешимое: предположим противное, рассмотрим

$$g(x) = \begin{cases} f(x), & x \in F \text{ – вычислимое, всюду определенное} \\ 0, & x \notin F \text{ – продолжение } f \end{cases}.$$

Замечание. $F = \{n \mid U(n,n) \text{ определено}\}$ — переформулировка L_1 (останавливающиеся на своем входе МТ.

Замечание. $\overline{F}$ — пример неперечислимого множества.

Замечание. Область определения универсальной функции — перечислимое, но не разрешмое. так как облатсь определения $U(n,n)$ — частично определенная не разрешимая.

Замечание. «Проблема остановки» — останавливается ли МТ M на входе x — переформулировка принадлежности области определения универсальной функции.

■

Теорема 2.3.5. Существует частичная вычислимая функция $f \colon \mathbb{N} \to \{0,1\}$, не имеющая всюду определенное вычислимое продолжение.

☐ Определим

$$d^{'''}(n) = \begin{cases} 1, & U(n,n) = 0 \\ 0, & U(n,n) > 0 \\ \uparrow, & U(n,n) \uparrow \end{cases}$$

Любое доопределение будет отличаться от $U(n,n)$ ■

Следствие 4. Существуют перечислимые X, Y, такие что $X \cap Y = \varnothing$, не отделимые никаким разрешимым множеством.

☐ $X = \{n \mid d^{'''}(n) = 0\}$ и $Y = \{n \mid d^{'''}(n) = 1\}$ Существует C разделяющее. ■

[1]далее это обозначение по умолчанию определяет универсальную вычислимую частичную фунукцию $U^{(m+1)}$ для вычислимых частичных функций m аргументов