

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

Тамарин Вячеслав

February 19, 2021

Contents

1	Вычислимость. Система вычислимости по Клини	3
1.1	Рекурсивные функции	3
1.1.1	Простейшие функции	3
1.1.2	Операторы	3
1.1.3	Функции	4
1.1.4	Оператор ограниченной минимизации	6
1.1.5	Предикаты	7
1.1.6	Теоремы про рекурсии	8
1.2	Равносильность МТ и ЧРФ	11
1.2.1	Функция Аккермана	14
2	Разрешимые и перечислимые множества	15
2.1	Определения	15

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

k -местная частичная функция, 3

кусочное задание функции, 10

общерекурсивная функция, 4

оператор минимизации, 4

оператор ограниченной минимизации, 6

оператор примитивной рекурсии, 3

оператор суперпозиции, 3

предикаты, 7

примитивно рекурсивная функция, 4

простейшие функции, 3

рекурсия возвратная, 9

рекурсия совместная, 10

частично рекурсивная функция, 4

Chapter 1

Вычислимость. Система вычислимости по Клини

1.1 Рекурсивные функции

Лекция 1: †

11 feb

Определение 1

Пусть функция $f: \mathbb{N}^k \rightarrow \mathbb{N}$, $k \in \mathbb{N}$, где $\mathbb{N} = \{0, 1, 2, \dots\}$. 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) — $I_n^m(x_1, \dots, x_n) = x_m$

1.1.2 Операторы

Определим три оператора:

Определение 2

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

$$h(y_1, \dots, y_m), g_i(x_1, \dots, x_n); 1 \leq i \leq n,$$

если

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

Оператор обозначается **S**.

- Функция $f^{(n+1)a}$ получается оператором примитивной рекурсии 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:

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

^aЗдесь и далее $f^{(n)}$ обозначается функция, принимающая n аргументов, то есть n -местная

^bподразумевается, что функция определена в этих точках

^cне определена

Пример 1.1.1.

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

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**:

$$I_1^1, s, I_3^3, h = S(s, I_3^3), f.$$

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. $x \dot{-} y = \begin{cases} 0 & x < y \\ x - y & \text{else} \end{cases}$
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.

$$5. \text{sg}(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases}$$

$$\begin{cases} \text{sg}(0) & = 0 \\ \text{sg}(x + 1) & = 1 = s(0) \end{cases}$$

6. Аналогично

$$7. f(x) = x \div 1$$

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

$$8. f(x, y) = x \div y$$

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

$$9. f(x, y) = |x - y| = (x \div y) + (y \div x)$$

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

$$\mu y \leq h(\bar{x})[g(\bar{x}, y) = 0]^a.$$

Это означает, что

$$f(\bar{x}) = \begin{cases} y & g(\bar{x}, y) = 0 \wedge y \leq h(\bar{x}) \wedge g(\bar{x}, i) \neq 0^b \forall i < y \\ h(\bar{x}) + 1 & \text{else} \end{cases}$$

^aЗдесь и далее $\bar{x} = x_1, \dots, x_n$.

^bАналогично, 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:

$$f(\bar{x}) = \sum_{y=0}^{h(\bar{x})} \prod_{i=0}^y \text{sg}(g(\bar{x}, i)).$$

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

Проверим, что можно получить $a(\bar{x}, y) = \sum_{i=0}^y g(\bar{x}, i)$ и $m(\bar{x}, y) = \prod_{i=0}^y g(\bar{x}, i)$ с помощью примитивной рекурсии:

$$\begin{cases} a(\bar{x}, 0) &= g(\bar{x}, 0) \\ a(\bar{x}, y+1) &= a(\bar{x}, y) + g(\bar{x}, y+1) \end{cases} \quad \begin{cases} m(\bar{x}, 0) &= g(\bar{x}, 0) \\ m(\bar{x}, y+1) &= m(\bar{x}, y) \cdot g(\bar{x}, y+1) \end{cases}$$

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).

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

Утверждение.

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



- Проверим, что характеристические функции примитивно / общерекурсивны:

$$\begin{aligned} \chi_{P \vee Q}(\bar{x}) &= \chi_P(\bar{x}) \vee \chi_Q(\bar{x}) \\ \chi_{P \wedge Q}(\bar{x}) &= \text{sg}(\chi_P(\bar{x}) + \chi_Q(\bar{x})) \\ \chi_{P \rightarrow Q}(\bar{x}) &= \overline{\text{sg}}(\chi_P(\bar{x}) + \overline{\text{sg}}(\chi_Q(\bar{x}))) \\ \chi_{\neg P}(\bar{x}) &= \overline{\text{sg}}(\chi_P(\bar{x})) \end{aligned}$$

- Аналогично выразим, через простейшие:

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

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



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

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

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

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

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

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

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

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 $\pi: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$:

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

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



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

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

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

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

Но $y \geq 0$, $x' \geq 0$, поэтому единственный возможный вариант, когда они равны нулю, а $x + y = x' + y' + 1$. Проверим на равенство $\pi(x, y)$ и $\pi(x', y')$:

$$\begin{aligned}\pi(x, y) &= \frac{1}{2}x(x+1) = \frac{1}{2}(y'+1)(y'+2) = \frac{1}{2}y'(y'+1) + y' + 1 \\ \pi(x', y') &= \frac{1}{2}y'(y'+1) + y'\end{aligned}$$

Равенства нет.

- 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:

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

Отсюда

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

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

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

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

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 $m(\bar{x}, y)$, которая возвращает закодированную последовательность $f(\bar{x}, i)$, $0 \leq i \leq y$.

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

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

$$f(\bar{x}, y) = \text{ith}(y, m(\bar{x}, y)).$$

Вернемся к построению m :

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



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

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

Если $g_i^{(n)}, h_i^{(k+2)}$, $1 \leq i \leq k$ — 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, \dots, c_k .

Давайте просто объединим все f_i в одну функцию

$$m(\bar{x}, y) = c(f_1(\bar{x}, y), \dots, f_k(\bar{x}, y)).$$

Теперь каждую f_i можно вычислить

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

Чтобы получить m достаточно использовать примитивную рекурсию:

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



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

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

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

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

^aНабор предикатов

^bТо есть для $i \neq j$ верно $R_i \cap R_j = \emptyset$.

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

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

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).

□

$2 \Rightarrow 1$ Если $f(x_1, \dots, x_n) = y$, то считаем, что МТ получаем $1^{x_1}01^{x_2}0 \dots 01^{x_n}$ и должна выдать 1^y ; если f не определена, МТ должна заикливиться 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:

$$(1^{x_1}01^{x_2} \dots 01^{x_n} *)^n.$$

- 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 псевдопараллельно (по очереди даем поработать).

В каждой части 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: †

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

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

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 $1^{x_1}0 \dots 01^{x_m}01^y$ построим $1^y01^{x_1}0 \dots 01^{x_m}001^{g(x_1, \dots, x_m)}$. 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 $1^y01^{x_1}0 \dots 01^{x_m}01^u01^z$ построим $1^y01^{x_1}0 \dots 01^{x_m}01^{u+1}01^{h(x_1, \dots, x_m, u, z)}$, 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 $1^y01^{x_1}0 \dots 01^{x_m}01^u0^z$ оставим только 1^z .
- Φ : для входа $1^y01^{x_1}0 \dots 01^{x_m}01^u01^z$ 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 .

М: Хотим по МТ M_g построить M_f , вычисляющую

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

Аналогично построим несколько вспомогательных МТ:

- N_1 : приписывает 0 ко входу:

$$1^{x_1}0 \dots 01^{x_m} \longrightarrow 1^{x_1}0 \dots 01^{x_m}0.$$

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

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

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

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

- N_5 : стирает все, кроме ответа

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

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

Теперь можем построить M_μ так:

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

1 \implies 2 Теперь мы хотим промоделировать работу МТ с помощью частично рекурсивной функции.

На вход должны либо выдать результат, либо заиклиться. 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

$$\tilde{\alpha}, \tilde{q}_i, \tilde{a}_j, \tilde{\beta},$$

где $\tilde{\alpha}$ — число, соответствующее α в m -ичной записи, \tilde{q}_i — просто номер состояния, \tilde{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 $\tilde{Q} = \{0, 1, \dots, |Q| - 1\}$.

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

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

$$\begin{aligned}\varphi_a: \tilde{Q} \times \tilde{\Gamma} &\rightarrow \tilde{\Gamma} \\ \varphi_q: \tilde{Q} \times \tilde{\Gamma} &\rightarrow \tilde{Q} \\ \varphi_d: \tilde{Q} \times \tilde{\Gamma} &\rightarrow \{1, 2\}\end{aligned}$$

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:

$$K = \alpha q_i a_j \beta \rightarrow \alpha' q'_i a'_j b' = K'.$$

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

Пусть

$$\begin{aligned}\tilde{q}'_i(\tilde{\alpha}, \tilde{q}_i, \tilde{a}_j, \tilde{\beta}) &= \varphi_q(\tilde{q}_i, \tilde{a}_j) \\ \tilde{\alpha}'(\tilde{\alpha}, \tilde{q}_i, \tilde{a}_j, \tilde{\beta}) &= \begin{cases} \tilde{\alpha} \cdot m + \varphi_a(\tilde{q}_i, \tilde{a}_j), & \varphi_d(\tilde{q}_i, \tilde{a}_j) = 1 \\ \left\lfloor \frac{\tilde{\alpha}}{m} \right\rfloor, & \varphi_d(\tilde{q}_i, \tilde{a}_j) = 2 \end{cases} \\ \tilde{\beta}'(\tilde{\alpha}, \tilde{q}_i, \tilde{a}_j, \tilde{\beta}) &= \begin{cases} \tilde{\beta} \cdot m + \varphi_a(\tilde{q}_i, \tilde{a}_j), & \varphi_d(\tilde{q}_i, \tilde{a}_j) = 1 \\ \left\lfloor \frac{\tilde{\beta}}{m} \right\rfloor, & \varphi_d(\tilde{q}_i, \tilde{a}_j) = 2 \end{cases} \\ \tilde{a}'_j &= \begin{cases} \tilde{\beta} \bmod m, & \varphi_d(\tilde{q}_i, \tilde{a}_j) = 1 \\ \tilde{\alpha} \bmod m, & \varphi_d(\tilde{q}_i, \tilde{a}_j) = 2 \end{cases}\end{aligned}$$

Заметим, что все эти формулы примитивно рекурсивные¹.

Общая работа МТ Пусть $K(0) = (\tilde{\alpha}_0, \tilde{q}_0, \tilde{a}_0, \tilde{\beta}_0)$ — начальная конфигурация. Чтобы получить новую конфигурацию для шага t , посчитаем все четыре параметра:

$$\begin{aligned}K(t) = (& \\ & K_\alpha(\tilde{\alpha}_0, \tilde{q}_0, \tilde{a}_0, \tilde{\beta}_0, t) \\ & K_q(\tilde{\alpha}_0, \tilde{q}_0, \tilde{a}_0, \tilde{\beta}_0, t) \\ & K_a(\tilde{\alpha}_0, \tilde{q}_0, \tilde{a}_0, \tilde{\beta}_0, t) \\ & K_\beta(\tilde{\alpha}_0, \tilde{q}_0, \tilde{a}_0, \tilde{\beta}_0, t) \\ &)\end{aligned}$$

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

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

$$\begin{cases} K_\alpha(\tilde{\alpha}_0, \tilde{q}_0, \tilde{a}_0, \tilde{\beta}_0, 0) &= \tilde{\alpha}_0 \\ K_\alpha(\tilde{\alpha}_0, \tilde{q}_0, \tilde{a}_0, \tilde{\beta}_0, t+1) &= \tilde{\alpha}'(K_\alpha(\dots, t), K_q(\dots, t), K_a(\dots, t), K_\beta(\dots, t)) \end{cases}$$

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 $\tilde{a}_0 + \tilde{b}_0 t \rightarrow a_z + \beta_z t$, если машина Тьюринга пришла сюда и не определена, если МТ закликивается:

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

Тогда результатом работы МТ будет

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

■

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. $\alpha_n(n): \mathbb{N} \rightarrow \mathbb{N}$ 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.

^aЭто 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).

□

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

$2 \implies 1$ 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:

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