Конспекты по дискретной матемтике

Анатолий Коченюк, Георгий Каданцев, Константин Бац

2022 год, семестр 4

Последний семестр дискретной математики. Две больших темы: производящие функции (комбинаторика) и введение в теорию вычислимости.

1 Производящие фукнции

Рассмотрим последовательности $\{a_n\}_{n\in\mathbb{N}}, \{b_n\}_{n\in\ltimes} \subset \mathbb{R}(\mathbb{C})$. Назовём эти последовательности A и B и будем почленную сумму обозначать кратко $A+B$. Это несколько неудобно и неестественно, об этих конвенциях нужно договариваться.

Вместо этого давайте рассмотрим формальный степенной ряд, у которого члены последовательности это коэффициенты ряда.

$$A(t) = a_0 + a_1 t + a_2 t^2 + \dots a_n t^n + \dots$$

Тогда почленная сумма последовательностей будет соотвествовать обычной сумме рядов $A(t) + B(t)$.

Чтобы сдвинуть последовательность на 1 вправо, можно просто умножить степенной ряд на x.

Можем рассмотреть степенной ряд-композицию $A(t^2) = a_0 + 0t + a_1 t^2 + \dots$. Это степенной ряд, соответвующий последовательности $a_0, 0, a_1, 0, a_2 \dots$.

Таким образом, мы можем "оперировать" над последовательностью как единым целым, и это очень удобно.

Мы не рассматриваем степенные ряды с стороны, с которой на них смотрит мат. анализ: как способ приблизить фукнцию, с некоторым радиусом сходимости и т.д. У нас степенные ряды *формальные* и не всегда (всегда не) должны пониматься как функции, в которой в переменную можно подставить значение.

$\mathbb{R}[x]$ — кольцо многочленов с коэффициентами из кольца R, состоящий из формальных многочленов. $\mathbb{R}[x]^+$ — множество формальных степенных рядов.

Определение 1. Формальный степенной ряд $A(t)$ последовательности $\{a_n\}_{n\in\mathbb{N}}$ называется производящей функцией (generating function).

Название неудачное. Оно связано с другими корнями понятия производящей функции (они нужны не только в комбинаторике).

Определена сумма производящих функций и произведение

$$A(t)B(t) = C(t) \quad c_n = \sum_{i=0}^{n} a_i b_{n-i}$$

Несмотря на то, что мы работаем с бесконечным по размеру обьектом, нам необходимо только конечное число элементов, чтобы посчитать каждый отдельынй его член. Этот раздел дискретной математики не любит предельных переходов.

Определено умножение на скаляр.

$$\lambda A(t) = C(t) \quad c_n = \lambda a_n$$

Определено даже деление!

$$\frac{A(t)}{B(t)} = C(t);\ b_0 \neq 0 \quad c_n = \frac{a_n - \sum\limits_{i=0}^{n-1} c_i b_{n-i}}{b_0}$$

Так можно посчитать, например, что

$$C(t) = \frac{1}{1-t} = 1 + t + t^2 + \dots t^n + \dots; \quad a_n = 1$$

Мы записали короткой (конечной) производящей функцией бесконечную последовательность. Более того, мы можем эту запись взять и производить с ней операции (умножать и складывать с другими производящими функциями).

$$\frac{1}{1-2t} = 1 + 2t + 4t^2 + \dots + 2^n t^n; \quad c_n = 2^n$$

Обобщая мы видим, что

$$\frac{1}{1-bt} = \sum_{n=0}^{\infty} b^n t^n = C(bt)$$

Вообще говоря,

$$A(t) = \sum a_n t^n \quad A(bt) = \sum a_n b^n t^n$$

Замечание. Если $b_0 = \pm 1$, $a_i, b_i \in \mathbb{Z}$, тогда $C = \frac{A}{B}$ с целочисленными коэффициентами $c_i \in \mathbb{Z}$.

$$\frac{1}{1-t-t^2} = 1 + t + 2t^2 + \dots + F_n t^n + \dots$$

Мы одной дробью породили целую последовательность Фиббоначи!

А как быть, если мы хотим взять последовательность и найти представление для её производящей функции? Мы можем поступить так.

$$F_0 = 1,\ F_1 = 1,\ F_n = F_{n-1} + F_{n-2}$$

Отсюда $F(t) + F(t)t = \frac{F(t)-1}{t}$ (следует из операций над производящими функциями и рекурентным соотношением последовательности фиббоначи).

А что с дифференцированием? Обыкновенная операция взятия производной (формального) степенного ряда позволяет нам умножать член последовательности на его номер.

$$A(t) \to A(t)' \cdot t$$

Эту операцию можно производить многократно, получая последовательноть членов исходной п-ти в k степени.

Как найти представление производящей функции для последовательности $a_n = n$?

$$a_n = n * 1$$

Производящая функция для п-ти единиц это $\frac{1}{1-t}$. Тогда

$$A(t) = \left(\frac{1}{1-t}\right)' t = \frac{t}{(1-t^2)}$$

Формальное деление это подтверждает.

А что с интегрированием? С интегрированием всё не очень мило.

А что с композицией?

$$C(t) = A(B(t))$$

Здесь много проблем доставляет свободый коэффициент у B. Давайте его уберем — $b_0 = 0$. Теперь мы можем посчитать

$$c_n = \sum_{k=0}^{n} a_k \sum_{n=i_1+i_2+\dots i_k} b_{i_1} b_{i_2} \dots b_{i_k}$$

Пример с доминошками.

Пример с деревьями.

1.1 Линейные рекуррентые последовательности. Регулярные производящие функции

Определение 2 (Линейные рекуррентные последовательности)**.** Пусть даны первые k членов последовательности $a_0, a_1, \ldots, a_{k-1}$. А все следующае члены определяются, как линейная комбинации k предыдущих.

$$a_n = a_{n-1} \cdot c_1 + a_{n-2} \cdot c_2 + \cdots + a_{n-k} \cdot c_k.$$

Такаяя последовательность называется **линейной рекурентной последовательностью**.

Пример. Числа Фибоначч. $F_0 = 1,\ F_1 = 1, \forall\ n \geqslant 2\ :\ F_n = F_{n11} + F_{n-2}$

$$\frac{1}{1-t-t^2} = \sum_{i=0}^{\infty} F_i t^i.$$

Обозначим $F(t) = \sum\limits_{i=0}^{\infty} F_i t^i = F_0 t^0 + F_1 t^1 + \sum\limits_{i=2}^{\infty} F_i t^i =$
$= 1 + t + \sum\limits_{i=2}^{\infty} F_{i-1} t^i + \sum\limits_{i=2}^{\infty} F_{i-2} t^i = 1 + t + t \cdot \sum\limits_{i=1}^{\infty} F_i t^i + t^2 \sum\limits_{i=0}^{\infty} F_i t^i =$
$= 1 + t + t \cdot (F(t) - 1) + t^2 \cdot F(t) \implies F = 1 + t \cdot F + t^2 F \implies$
$F(t) = \dfrac{1}{1-t-t^2}.$

Теорема 1. Пусть есть линейная рекуррентная последовательность порядка k:
$a_0, a_1, \ldots, a_{k-1}, \ldots.$

Даны $a_0, \ldots, a_{k-1},\ \forall n \geqslant k\ :\ a_n = \sum\limits_{i=1}^{k} a_{n-i} \cdot c_i.$

Тогда $A(t) = \sum\limits_{i=0}^{\infty} a_i t^i = \dfrac{P(t)}{Q(t)}$ — рациональная функция, где
$Q(t) = 1 - c_1 t - c_2 t^2 - \cdots - c_k t^k$, а $P(t) = \ldots.$

Доказательство. Обозначим

$$A(t) = \sum_{i=0}^{\infty} a_i t^i = \sum_{i=0}^{k-1} a_i t^i + \sum_{n=k}^{\infty} a_i t^n.$$

Сразу заменим последнюю сумму предположеним из теоремы, получим

$$A(t) = \sum_{i=0}^{k-1} a_i t^i + \sum_{n=k}^{\infty} t^n \sum_{i=1}^{k} a_{n-i} \cdot c_i = S + \sum_{i=1}^{k} c_i \sum_{n=k}^{\infty} a_{n-i} t^n = S + \sum_{i=1}^{k} c_i \cdot t^i \cdot \sum_{n=k-1}^{\infty} a_n t^n =$$

$$= S + \sum_{i=1}^{k} c_i \cdot t^i \cdot (A(t) - A_{k-1}(t)) = X.$$

Пусть $C(t) = \sum\limits_{k=1}^{k} c_i t^i$, тогда $Q(t) = q - C(t)$. $\quad X = S + C(t) \cdot A(t) - \sum\limits_{k=1}^{k} c_i t^i A_{k-i}(t) = Y.$

Пусть $F(t) \,\%\, t^k = \sum\limits_{i=0}^{k-1} f_i t^i$, тогда $A_{k-i}(t) = A(t) \,\%\, t^{k-1}$.

$$\sum_{k=1}^{k} c_i t^i A_{k-i}(t) \cdot Ak - i(t) = A(t) \,\%\, t^{k-i} = (C(t) \cdot A(t)) \,\%\, t^k \implies$$

$$A(t) = \sum_{i=0}^{k-1} a_i t^i + C(t) \cdot A(t) - (C(t) \cdot A(t)) \,\%\, t^k \implies A(t)(1 - C(t)) = ((1 - C(t)) \cdot A(t)) \,\%\, t^k$$

$$\Longrightarrow A(t) = \frac{P(t)}{Q(t)}, \quad \text{где} \quad Q(t) = 1 - C(t) = 1 - c_1 t - c_2 t^2 - \dots - c_k t^k,$$

$$P(t) = \left(\left(\sum_{i=0}^{k-1} a_i t^i \right) \cdot Q(t) \right) \mod t^k.$$

■

Пример. Для чисел фибоначчи: $a_0 = a_1 = 1,\ c_1 = c_2 = 1 \Longrightarrow$

$$A(t) = \frac{(1+t)\cdot(1-t-t^2) \mod t^2}{1-t-t^2}.$$

$$a_0 = 6,\ a_1 = -3,\ c_1 = c_2 = 1 \quad \Longrightarrow \quad A(t) = \frac{(6-3t)\cdot(1-t) \mod t^2}{1-t-t^2} = \frac{6-9t}{1-t-t^2}.$$

Доказательство в обратном направлении. Частный случай:

$$\frac{1}{1-C(t)} = A(t),\ A(t)\cdot(1-C(t)) = 1,$$

$$t^0 = a_0 = 1,\ t^1 : a_1 \cdot 1 - a_0 c_1 = 0,\ t^2 : a_2 \cdot 1 - a_1 \cdot c_1 - a_0 c_2 = 0.$$

Посмотрим на некоторую производящую функцию, например $\frac{1-3t+6t^3}{1-t-t^2-t^4}$. Понимаем, что $a_n = a_{n-1} + a_{n-2} + a_{n-4}$.

$$a_0 = 1,\ a_1 = 1-3 = -2,\ a_2 = 1-2 = -1,\ = -1-2+6 = 3$$

$$A(t)\cdot Q(t) = P(t). \quad \sum_{i=0}^{n} q_i \cdot a_{n-i} = p_n \quad a_n = p_n - \sum_{i=1}^{k} q_i \cdot a_{n-i}.$$

■

Пусть $a_0, a_1, \dots, a_{k-1},\ \forall n \geqslant k\ : a_n \sum\limits_{i=1}^{k} a_{n-i} \cdot c_i$.

Задача: посчитать a_n.
Можно явно за $\mathcal{O}(n \cdot k)$.
Можно через возведение матрицы в степень за $\mathcal{O}(k^3 \log_2 n)$.
Потом мы научимся делать это за $\mathcal{O}(k^2 \log_2 n)$.

На самом деле, для одной и той же числовой последовательности можно получить несколько производящих функций.

$$A(t) = \frac{P(t)}{Q(t)} \cdot \frac{Q(-t)}{Q(-t)} = \frac{P(t)\cdot Q(-t)}{Q(t)\cdot Q(-t)}.$$

Например, для чисел Фибоначчи:

$$\frac{1}{1-t-t^2} \cdot \frac{1+t+t^2}{1+t+t^2} = \frac{1+t-t^2}{1-3t^2+t^4}. \quad F_n = F_{n-2}\cdot 3 - F_{n-4}.$$

Теорема 2. Для производящих функций, задающих рекурентные соотношения эквивалентны следующие высказывания

- $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \dots +$
- $A(t) = \frac{P(t)}{Q(t)}$
- $a_n = \sum\limits_{i=1}^{b} p_i(n) \cdot r^i$, где $r_i \in \mathbb{C}$

$Q(t) = 1 - c_1 t - c_2 t^2 - \dots - c_k t^k$

$P(t)$ определяет то, как надо подправить первые члены, чтобы получились те, которые нужны.

$A(t)Q(t) - P(t) = 0$.

Как посчитать r?

Пусь $Q(t) = 1 - rt$,

$a_n = r \cdot a_{n-1}$

$a_m = r \cdot a_{m-1}$

$a_{m+1} = r \cdot a_m$

$\dots$

$a_n = r^n \cdot \dfrac{a_{m-1}}{r^{m-1}}$

Пусть $Q(t) = (1 - r_1 t)(1 - r_2 t), \;\; r_1 \neq r_2$.

Лемма 2.1. $Q(t) = \prod\limits_{i=1}^{n} (1 - r_i t)$, где $r_i \neq r_j$ $\dfrac{P(t)}{Q(t)} = \sum\limits_{i=1}^{n} \dfrac{P_i(t)}{1 - r_i t}$

$Q(t) = \sum\limits_{i=1}^{n} d_i r_i^n$.

r_i — числа, обратные корням многочлена Q. Если степень Q равно k, то Q имеет ровно k корней (с учетом кратности).

Таким образом, $Q(t) = q_k \prod\limits_{i=1}^{k} (t - t_i) = (-1)^k q_k \prod\limits_{i=1}^{k} \left(1 - \dfrac{t}{t_i}\right) \cdot t_i =$

$= \left[(-1)^k q_k \cdot \prod\limits_{i=1}^{k} t_k\right] \prod\limits_{i=1}^{k} (1 - r_i t) = \alpha \prod\limits_{i=1}^{k} (1 - R_i t)$.

Почему нет корня 0? Потому что $Q(t)$ имеет вид $Q(t) = 1 - c_1 t - \dots$.

Пример. Рассмотрим числа Фибоначчии. $F(t) = \dfrac{1}{1 - t - t^2}$.

Корни $t_{1,2} = \dfrac{-1 \pm \sqrt{1+4}}{2}$, обратные корини $r_{1,2} = \dfrac{1 \mp \sqrt{5}}{2}$. Обратные корни разные — нам очень приятно.

$$Q(t) = \left(1 - \frac{1 - \sqrt{5}}{2}\right)\left(1 - \frac{1 + \sqrt{5}}{2}\right).$$

$$\frac{1}{(1 - r_1 t)(1 - r_2 t)} = \frac{c_1}{1 - r_1 t} + \frac{c_2}{1 - r_2 t}, \quad c_1(1 - r_2 t) + c_2(1 - r_1 t) = 1 \implies$$

$$\begin{cases} c_1 + c_2 = 1 \\ c_1(-r_2) + c_2(-r_1) = 0 \end{cases} \implies c_2 = \frac{-r_2}{r_1 - r_2} = \frac{-1 - \sqrt{5}}{2 \cdot (-\sqrt{5})} = \frac{5 + \sqrt{5}}{10}, \quad c_1 = \frac{5 - \sqrt{5}}{10}.$$

$$a_n = c_1 r_1^n = \frac{5 - \sqrt{5}}{10} \cdot \left(\frac{1 - \sqrt{5}}{2}\right)^n b_n = c_2 r_2^n = \frac{5 + \sqrt{5}}{10} \cdot \left(\frac{1 + \sqrt{5}}{2}\right)^n \implies$$

$$f_n = \frac{5 - \sqrt{5}}{10} \cdot \left(\frac{1 - \sqrt{5}}{2}\right)^n + \frac{5 + \sqrt{5}}{10} \cdot \left(\frac{1 + \sqrt{5}}{2}\right)^n = \Theta\left(\left(\frac{1 + \sqrt{5}}{2}\right)^n\right).$$

Замечание. Если λ — единсвенный минимальный по модулю комплексный корень $Q(t)$, $A(t) = \dfrac{P(t)}{Q(t)}$, то $a_n = \Theta\left(\dfrac{1}{\lambda^n}\right)$.

$\dfrac{1}{(1 - rt)^2} = \dfrac{1}{1 - 2rt + r^2 t^2}$. $a_0 = 1, \;\; a_1 = 2r, \;\; a_2 = 3r^2, \; a_3 = 4r^3, \quad \dots, \; a_n = (n+1) r^n$.

$\dfrac{1}{r}\left(r^n t^n\right)' = \dfrac{1}{r} \sum n r^n t^{n-1} = \sum n r^{n-1} t^{n-1} = \sum (n+1) r^n t^n$.

Лемма 2.2. $\frac{1}{1-rt}^s = \sum\limits_{n=0}^{\infty} p_s(n) r^n t^n$

Доказательство. Докажем по индукции.

1. База. $s = 0$ — просто

2. Переход. Далее много формул. $\left(\frac{1}{(1-rt)^s}\right)' = \frac{-r(-s)}{(1-rt)^{s+1}}$. $\left(\sum\limits_{n=0}^{\infty} p_s(n) r^n t^n\right)' = \sum\limits_{n=1}^{\infty} n p_s(n) r^n t^{n-1} = \sum\limits_{n=0}^{\infty} (n+1) p_s(n+1) r^{n+1} t^n$. $\frac{1}{(1-rt)^{s+1}} = \sum\limits_{n=0}^{\infty} \frac{n+1}{s} p_s(n+1) r^n t^n$, $p_{s+1}(n) = p_s(n+1)\frac{n+1}{s} = \sum\limits_{i=0}^{s-1} p_{s,i}(n+1)^i \frac{n+1}{s}$.

$$p_{s,i} = \frac{a_{s,i}}{b}, \quad b = s!,\ a_{s,i} \in \mathbb{Z}$$

■

Теорема 3. Пусть $A(t) = \dfrac{P(t)}{Q(t)}$, r_i — обратный корень кратности s_i Q_i, количество различных корней b. Тогда начиная с некоторого места (но точно, начиная с k) $a_n = \sum\limits_{i=1}^{b} p_i(n) r_i^n$, $\quad \deg p_i = s_i - 1$, $\quad \sum\limits_{i=1}^{b} s_i = k$.

Доказательство.

$$Q(t) = \prod_{i=1}^{b} (1 - r_i t)^{s_i}, \quad \frac{P(t)}{Q(t)} = \sum_{i=1}^{b} \frac{P_i(t)}{(1 - r_i t)^{s_i}}.$$

■

Если λ_i — единственный минимальный комплексный корень $Q(t)$ кратности s_i. Тогда $a_n = \Theta\left(\dfrac{n^{s_i-1}}{\lambda_i^n}\right)$.

Пример. $a_n = n^3$, $\quad a_n = 4 \cdot a_{n-1} - 6 \cdot a_{n-2} + 4 \cdot a_{n-3} - a_{n-4}$.

Подберем поправку первых членов: $P(n) = t + 4t^2 + t^3$.

Утверждение 3.1. Асимптотическое поведение рекуррентности не зависит от начальных значений, оно зависит только от коэффициентов соотношений.

Утверждение 3.2. Пусть $\lambda_1, \lambda_2, \ldots, \lambda_z$ — минимальные корни максимальной кратности.

$\lambda_j = \dfrac{e^{i\varphi_j}}{r}$. $\varphi_j = \dfrac{p_j}{q_j} \cdot 2\pi$.

Пусть $\overline{q} = LCM(q_j)$. Тогда последовательность a_i имеет асимптотическое поведение при $i \% \overline{q} = const$.

1.2 Комбинаторика и производящие функции

Пример. Замещение прямоугольника $2 \times n$ доминошками вида 1×2.

$$\frac{1}{1 - t - t^2} = 2 + t + t^2 + t^2 + t^3 + t^3 + t^3 + \ldots = 1 + t + 2t^2 63t^3 + \ldots + F_n t^n$$

Комбинаторные обьекты это конструкции, которые состоят из атомов и разных связей атомов между собой. Под атомом мы понимаем некоторую неделимую часть комб. обьекта. Давайте все наши комбинаторные обьекты сложим.

В этой сумме заменим каждый атом на t^ω, где ω — вес данного атома. Потом t^ω атомов одного обьекта перемножим.

Вес обьекта — сумма весов его атомов.

Пример. A – множество комбинаторных объектов. Давайте их просуммируем.

$\Delta_1 + \Delta_2 + \Delta_3 + \ldots$

атом (неделимое) – то, что мы считаем.

$t^{\omega(\Delta_1)} + t^{\omega(\Delta_2)} + t^{\omega(\Delta_3)} + \ldots = \sum\limits_{n=0}^{\infty} a_n t^n = A(t)$ — производящая функция для объектов веса t.

Базовые комбинаторные объеты

Определение 3 (Базовые объекты). $U = \{u\} \quad \omega(u) = 1 \quad u(t) = t$ – производящая функция для этих комбинаторных объектов

$B = \{a, b\} \quad \omega(a) = \omega(b) = 1 \quad B(t) = 2t$

$E = \{\varepsilon\} \quad \omega(\varepsilon) = 0 \quad E(t) = 1$

$E_k = \{\varepsilon_1, \varepsilon_2, \dots, \varepsilon_k\} \quad E_k(t) = k$

$D = \{a, A\} \quad \omega(a) = 1 \quad \omega(A) = 2 \quad D(t) = t + t^2$

Операции конструирования

Определение 4 (Дизъюнктное объединение). A, B – множества комбинаторных объектов и $A \cap B = \varnothing$

Пусть $C = A \cup B$. Тогда производящая функция

$$\begin{aligned} C(t) &= A_1 + A_2 + \dots + B_1 + B_2 + \dots \\ &= t^{\omega(A_1)} + t^{\omega(A_2)} + \dots + t^{\omega(B_1)} + t^{\omega(B_2)} + \dots \\ &= A(t) + B(t) \end{aligned}$$

.

Определение 5 (Упорядоченная пара (прямое произведение)). Пусть $A, B, \quad A(t), B(t)$ — их производящая функция. Определим пару C, как $A \times B = C = \{\langle a, b\rangle \mid a \in A, b \in B\}$.

$C_n = C \cap \{x \mid \omega(x) = n\} \quad \langle a, b\rangle\, \omega(a) = i \quad \omega(b) = j \quad i + j = n \quad j = n - i$.

$C_n = \cup A_i \times B_{n-i}$. $c_n = \sum\limits_{i=0}^{n} a_i \cdot b_{n-i} \implies C(t) = A(t) \cdot B(t)$.

Замечание (Комбинаторный мысл прямого произведения). Пусть у нас есть объекты $A = A_1 + A_2 + \dots + A_k + \dots, \quad B = B_1 + B_2 + \dots + B_k + \dots$.

$(A_1 + A_2 + \dots) \cdot (B_1 + B_2 + \dots) = A_1 \cdot B_1 + A_1 \cdot B_2 + \dots + A_2 \cdot B_1 + A_2 \cdot B_2 + \dots$.

$\langle a, b\rangle \quad t^{\omega(a)} t^{\omega(b)} = t^{\omega(a) + \omega(b)}$.

Определение 6 (Последовательность (sequence)). Определим последовательность из A, как $SeqA = \{[], [A_1], [A_2], \dots, [A_1, A_2], [A_1, A_3], \dots\}$.

$SeqA = [] \cup A_1 \cdot ([] + [A_1] + [A_2] + \dots) + A_2 \cdot ([] + \dots) = 1 + A \times SeqA$.

$B = SeqA \implies B(t) = 1 + A(t)B(t) \implies B(t) = \frac{1}{1 - A(t)}$.

Определение 7 (Последовательность (sequence), второй способ). $SeqA = A^0 \cup A^1 \cup A^2 \cup \dots \cup A^k \cup \dots$

A^i – декартова степень, последовательности длины i

$$B(t) = A(t)^0 + A(t)^1 + A(t)^2 + \dots + A(t)^k + \dots = \frac{1}{1 - A(t)}$$

Пример. $SeqU = \{[], [u], [u, u], [u, u, u], \dots\} = N$. $n_k = 1, \ U(t) = 1 \implies SeqU = \dfrac{1}{1 - t}$.

$SeqB = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, \dots\} = C$. $c_n = 2^n, \quad B(t) = 2t \implies C(t) = \dfrac{1}{1 - 2t}, \quad c_n = 2 \cdot c_{n-1}$.

$C_n = 2^n \quad B(t) = 2t \quad C(t) = \frac{1}{1 - 2t}$

$C_n = 2C_{n-1}$

$SeqE = \{[], [\varepsilon], [\varepsilon, \varepsilon], \ldots\} = C$
$E(t) = 1 \quad C(t) = \frac{1}{1-E(t)} = \frac{1}{1-1} = \frac{1}{0}$ ☹
$C_0 = +\infty??$

Пример. $C = SeqD = \{\varepsilon, a, aa, aA, A, Aa, AA, \ldots\}$
$C(t) = \frac{1}{1-D(t)} = \frac{1}{1-t-t^2}$
a – одна вертикальная доминошка, вес 1. A – две горизонтальные доминошки, вес 2.
$C = aC + AC \quad C(t) = tC(t) + t^2C(t)$

Определение 8 (Множество)**.** Обозначается Set или $PSet$.
$B = \{a, A\} \quad SetB = \{\varnothing, \{a\}, \{A\}, \{a, A\}\}$.
$C = SetA$.
$a \in A \quad B_a = \varepsilon + a$ – либо берём, либо не берём. C – дкартово произведение по всем a.
$C(t) = \prod\limits_{a\in A}\left(1 + t^{\omega(a)}\right) = \prod\limits_{n=0}^{\infty}(1+t^n)^{a_n}$.

Пример. Возьмем $U = \{u\}$. $SetU = C = \{\varnothing, \{u\}\}$. Найдем $C(t)$.

$$C(t) = \prod_{n=0}^{+\infty}(1+t^n)^a_n = (1+t)^1 = 1+t.$$

Пусть $B = \{a, A\}$, $\;C = SetB$. Заметим, что $b_1 = 1, b_2 = 1$.

$$C(t) = \prod_{n=0}^{+\infty}(1+t^n)^{b_n} = (1+t)(1+t^2) = \underbrace{1}_{\varnothing} + \underbrace{t}_{a} + \underbrace{t^2}_{A} + \underbrace{t^3}_{a,A}.$$

$\prod\limits_{n=0}^{\infty}(1+t^n)^{a_n} = (a+t_0)^{a_0} \cdot \prod\limits_{n=1}^{\infty}(1+t^n)^{a_n} = 2^{a_0}\prod\limits_{n=1}^{\infty}(1+t^n)^{a_n}$.

Определение 9 (Мультимножество)**.** Обозначается $MSetA$.
Мы можем включить каждый объект $0, 1, 2, \ldots$.
$\varepsilon + a + aa + \ldots = Seq\{a\}$.
$a_1 \in A,\ a_2 \in A \implies Seq\{a_1\} \times Seq\{a_2\} = MSet\{a_1, a_2\}$.
$MSetA = \prod\limits_{a\in A} Set\{a\}$.

$$C(t) = \prod_{a\in A} Seq\{a\} = \prod_{a\in A}\frac{1}{1-t^{\omega(a)}} = \prod_{n=1}\left(\frac{1}{1-t^n}^{a_n}\right) = \prod_{n=1}^{\infty}(1-t^n)^{-a_n}.$$

Пример. $U = \{u\} \quad C = MSetU \quad C(t) = \prod\limits_{n=1}^{\infty}(1-t^n)^{-u_n} = (1-t)^{-1} = \frac{1}{1-t}$.
$B = \{a, A\} \quad C = MSetB \quad b_1 = 1 = b_2$.
$C(t) = \prod\limits_{n=1}^{\infty} = (1-t)^{-1}(1-t^2)^{-1} = \frac{1}{(1-t)(1-t^2)}$.
Ассимптотика C_n.
$Q(t) = (1-t)(1-t^2) = (1-t^2)(1+t)$.
Корни: $t = \pm 1$. Обратные корни $r = \pm 1$ Кратность $r_1 = 1 \quad s_1 = 2 \qquad r_2 = -1 \quad s_2 = 1$.
$(a_n + b)\cdot 1^n + c\cdot(-1)^n$
$\frac{1}{2}n + \frac{1}{2}(-1)^n + const$

1.2.1 Циклы (cycle)

$B = \{a, b\} \quad CycB = \{\varepsilon, a, b, ab, aa, bb, aaa, aab, abb, bbb, aaaa, aaab, aabb, abab, abbb, bbbb, \ldots\}$
Раньше мы называли такие комбинаторные объекты *ожерельями*.

$C = CycB = \bigcup_{k=1}^{\infty} (CycA)_k$.

$C(t) = \sum\limits_{k=1}^{\infty} C_k(t), \quad C_k(t)$ — производящая функция длинны k.

$C_k(t)$ – последоватльности длины k с точностью до циклического сдвига.

S_k – последовательности длины $k \quad = (A(t))^k \quad C_k(t)/G \quad G$ – группа циклических сдвигов.

$C_{k,n} = \frac{1}{k} \cdot \sum\limits_{i=0}^{k-1} |I(i)|$.

Количество классов эквивалентности по лемме Бёрнсайда равно $\gcd(i,k)$. Внутри класса одинаковые объекты. Размер класса $\frac{k}{\gcd(i,k)}$.

n кратно $\frac{k}{\gcd(i,k)}$. $S_{\gcd(i,k)} \frac{n \cdot \gcd(i,k)}{k}$.

$$C_{n,k} = \frac{\sum\limits_{j=0}^{k-1} S_{\gcd(i,k), \frac{n \cdot gcd(i,k)}{k}}}{k}$$

2 Формальные языки

Как объяснить комьютеру, что я вляется словом в нашем языке.

Пусть $L \subset L^*$. Мы знаем два способа задания языка.

1. ДКА = Р.В.
2. КСГ = МП-автомат.

А что мы вообще сделать с помощью компьютера? Ну что-то нельзя сделать из-за фикики: путешествовать во времени, ... А что нельзя сделать с помощью математики?

Есть два способа рассказать компьютеру о языке:

- Научить его распознавать слово из языка.

  Например, ДКА. Нужен мета-язык описания конечных автоматов и само описание автомата.

- Научить компьютер пораждать слова из языка.

  У нас есть также мета-язык описания пораждения языка и само описание.

  Например, даем компьютеру парсер регулярных выражений, само регулярное выражение, компьютер строит дерево разбора и генерирует нам слова.

На сколько сильно мы можем усложнить граматику нашего языка. Регулярные языки не умели генрировать палиндромы, КСГ не умели генерировать $1^n 2^n 3^n$. А на сколько еще мы можем усложнить наши описания?

Вообще, языков может быть 2^{Σ^*} — несчетное количество. Но для нас это не страшно, почти все из них описать не возможно. А что можно описать? Ну пусть то, что умеет понимать компьютер. **А что такое вообще компьютер?**

Для осознания мощности компьютеров существуют модели. Мы будем изучать **Машину Тьюринга**. Машина Тьюринга основывается на ленте. Так же существует, например, машина Маркова, там немного другой принцип.

Метод описания

Пусть у нас есть *соременный x86* компьютер. Как описать модель такого комьютера? Ну есть почти неограниченная память и есть какие-то операции (сложение, умножение, вызов функции). Что такое программа для такого компьютера? Программа — $\underbrace{\quad}_{\text{строка}} \in \Pi^*$. На самом деле можно записать в битовом формате $\mathbb{B} = \{0,1\}$, то есть $\underbrace{\quad}_{\text{Description}} \in \mathbb{B}^*$. На самом деле, описание программ самый мощный инструмен. Пусть есть описание конченого автомата, запишим его в константу, добавим код имплементации конечного автомата, получим описание языка от автомата на языке программ.

Утверждение 3.3 (Тезиси Тьюринга–Чёрча)**.** Все, что можно выразить на «обычном компьютере» можно выразить на Машине Тьюринга.

Замечание. Почему это не утверждение или теорема? Надо доопределить понятие «обычный комьютер» и тогда получится какое-то утверждение.

А что вообще может выдавать программа? *Comilation error* не больная проблема, давайте в таком случае программа будет делать что-то конкретное, например No. Если *Runtime error*, путь тоже будем возвращать что-то конкретное. *Memory limit* — не проблема, попросим пользователя добавить память или подождем, пока изобретут комьютер лучше. Самое интересное — *Time limit*, это обозначают $\bot$. Таким, образом наша программа будет возвращать три значения $И, Л, \bot$. Тогда хочется добавить ограничение на время исполненя, правда, это немного ограничит класс зязыков.

Определение 10. Язык L разрешимый (рекурсивный), если $\exists$ программа p $\forall x \in L \implies p(x) = 1, \quad x \notin L \implies p(x) = 0$.

Определение 11. Язык L полуразрешимый (перечислимый, рекурсивнно перечислимый), если $\exists$ программа p $\forall x \in L \iff p(x) = 1$.

На самом деле полуразрешимые описания языков — мкисмальные по мощности. Разрешимый — максимальный по мощности прикладной способ описания.

Существуют не разрешимые и не полуразрешимые языки.

Метод порождения

Пусть у нас есть компьютер, который по опсанию выводит список слов. Можно выводить первые n слов.

Опять же, понятно, что описание с помощью компьютера макисмальное по мощности.

Определение 12. Язык L перечеслимым, если можно написать программу, которая выодит его слова.

Теорема 4. L полуразрешим $\iff$ L перечислим.

Определение 13. Градуированный лексикографический порядк — перечисление в порядке увеличения длинны, а среди слов с равной длинны лексикографически.

Доказательство. $\implies$

Неверный подход 1.

```
for (x ∈ Σ*)
    if p(x):
        print(x)
```

Не верный поход, попытка 2

```
for (TL = 1; True; TL++)
    for (x ∈ \Sigma^{*})
        if p |_T (x):
            print(x)
```

Подход правильный:

```
for (TL = 1; True; TL++)
    for (x ∈ Σ*[: TL])
```

```
            if p |_T (x):
                if (x ∉ was):
                    print(x)
                    was.add(x)
```

$\Longleftarrow$

Пусть у нас есть q — перечислитель L.

```
p(x):
    while q.next() != x:
        pass
    return True
```

■

Пример непреичлимого языка

Программа набор из 0 и 1. Пусть A — предикат. $L_A = \{p \mid A(p)\}$ — формальный язык.

Определение 14 (Универсальный язык)**.** $U = \{\langle p, x\rangle \mid p(x) = 1\}$.

Замечание. U — полуразрешим.

Доказательство. Давайте сделаем

```
inU(⟨p,x⟩):
    return p(x)
```

■

Теорема 5. U не разрешим.

Доказательство. Пусть есть функция $\text{inU}(\langle p, x\rangle)$ — всегда завершается.

```
q(x):
    if inU(⟨x,x⟩):
        return 0
    else:
        return 1
```

Посчитаем q(q).

Если $\text{q(q)} = 1 \implies \text{inU}(\langle q, q\rangle) = \text{false} \implies \text{q(q)} = 0$ (плохо).

Пусть $\text{q(q)} = 0 \implies \text{inU}\ (\langle q, q\rangle) = \text{true} \implies \text{q(q)} = 1$ (тоже плохо).

■

Теорема 6. Если A и $\overline{A}$ — полуразрешим $\implies$ A — разрешим.

Утверждение 6.1. A разрешим $\implies$ $\overline{A}$ — разрешим.

Доказательство теоремы. .

```
inA(x):
    for (TL = 1; +∞):
        if p |_TL(x):
            return 1
        if q |_TL(x):
            return 0
```

■

Не полуразрешим

Пример. Язык дополнение до U не полуразрешим.

На самом деле эти множества биективны:

- Строки Σ^*
- Программы Prog
- Числа $\mathbb{N}^+$

$\mathrm{Prog} \underset{\mathrm{id}}{\iff} \Sigma^* \underset{\mathrm{grad\,order}}{\iff} \mathbb{N}^+$.

- Полуразрешимые языки $\iff$ вычислимые функции $A \subset \mathbb{N} \to \{0, 1\}$.
- Разрешимые языки $\iff$ всюду определенные (Hall) вычислимые функции $\mathbb{N} \to \{0, 1\}$.