

# КОЛЛОК 4

Cosmo Chief

10 июня 2025 г.

## Содержание

1 Formal languages	3
2 Operations on formal languages	3
3 Regular languages	3
4 Closure properties of regular languages	3
5 Pumping lemma	4
6 Regular expressions	4
7 Deterministic finite automata (DFA)	5
8 Non-deterministic finite automata (NFA)	5
9 Rabin-Scott powerset construction	5
10 Epsilon-NFA	6
11 NFA construction from epsilon-NFA	6
12 Kleene's theorem	6
13 Kleene's algorithm	6
14 Thompson's construction	7
15 Ordered arrangements	7
16 Permutations	7
17 k-permutations	7
18 Cyclic permutations	7
19 Unordered arrangements	8
20 k-combinations	8
21 Binomial theorem	8
22 Multisets	8
23 Permutations of multisets	8
24 Combinations of infinite multisets	8

25 Multinomial theorem	9
26 Compositions	9
27 Set partitions	9
28 Stirling numbers of the second kind	9
29 Bell numbers	10
30 Integer partitions	10
31 Principle of Inclusion-Exclusion	10
32 Recurrence relations	10
33 Solving recurrence relations using characteristic equations	11
34 Generating functions	11
35 Solving linear recurrences using generating functions	11
36 Solving combinatorial problems using generating functions	11
37 Operators and annihilators	12
38 Solving linear recurrences using annihilators	12
39 Catalan numbers	12
40 Generalized binomial theorem	12
41 Gamma function	13
42 Divide-and-Conquer algorithms analysis using recursion trees	13
43 Master theorem	13
44 Akra–Bazzi method	13

# 1 Formal languages

Формальный язык  $\mathcal{L}$  - некоторое подмножество всех слов из заданного алфавита  $\Sigma$ .

$$\mathcal{L} \subseteq \Sigma^* \quad \Sigma^* = \sum_{k=0}^{\infty} \Sigma^k$$

## 2 Operations on formal languages

- Как множества:
  - Объединение  $\mathcal{L}_1 \cup \mathcal{L}_2$
  - Пересечение  $\mathcal{L}_1 \cap \mathcal{L}_2$
  - Дополнение  $\neg \mathcal{L} = \Sigma^* \setminus \mathcal{L}$
  - ...?
- Конкатенация  
 $\mathcal{L}_1 \cdot \mathcal{L}_2 = \{x + y \mid x \in \mathcal{L}_1, y \in \mathcal{L}_2\}$
- Возведение в степень  
 $\mathcal{L}^k = \underbrace{\mathcal{L} \cdot \dots \cdot \mathcal{L}}_{k \text{ раз}}$   
 $\mathcal{L}^0 = \{\varepsilon\}, \varepsilon = \text{пустое слово}$
- Звезда Клини  
 $\mathcal{L}^* = \bigcup_{k=0}^{\infty} \mathcal{L}^k$

## 3 Regular languages

Множество регулярных языков задается так:

$$\begin{aligned} REG &= \bigcup_{k=0}^{\infty} Reg_k = Reg_{\infty} \\ Reg_{i+1} &= Reg_i \cup \{\mathcal{L}_1 \cup \mathcal{L}_2, \mathcal{L}_1 \cdot \mathcal{L}_2, \mathcal{L}_1^* \mid \mathcal{L}_1, \mathcal{L}_2 \in Reg_i\} \\ Reg_0 &= \{\emptyset, \varepsilon, \{c\} \mid c \in \Sigma\} \end{aligned}$$

Какой-то конкретный регулярный язык - регулярный язык, который может быть распознан каким-то конечным автоматом

**Теорема Клини** утверждает, что класс регулярных языков совпадает с классом языков, распознаваемых конечными автоматами

## 4 Closure properties of regular languages

Множество регулярных языков замкнуто относительно

- Объединения
- Конкатенации
- Звезды Клини

Очевидно из определения:

$$Reg_{i+1} = Reg_i \cup \{\mathcal{L}_1 \cup \mathcal{L}_2, \mathcal{L}_1 \cdot \mathcal{L}_2, \mathcal{L}_1^* \mid \mathcal{L}_1, \mathcal{L}_2 \in Reg_i\}$$

Помимо этого, множество регулярных языков замкнуто относительно

- Пересечение  $\mathcal{L}_1 \cap \mathcal{L}_2$
- Positive closure  $\mathcal{L}^+$   
По сути это все та же звезда Клини, но язык сконкатенирован сам с собой  $k \geq 1$  раз. Тоест  $\mathcal{L}^+ \subseteq \mathcal{L}^*$
- Дополнение  $\neg \mathcal{L}$
- Разность  $\mathcal{L}_1 \setminus \mathcal{L}_2$
- Reversal  $\mathcal{L}^R$   
По сути - берете конечный автомат, который выражает этот язык, и разворачиваете все ребра. (наверное надо будет еще добавить входную вершину, и из нее  $\varepsilon$ -переходы в каждую, ранее терминальную вершину, а ранее входная вершина станет терминальной)  
Еще проще  
 $\forall w \in \mathcal{L} \mid w[:: -1] \in \mathcal{L}^R$   
Впрочем, скорее даже  
 $\{w[:: -1]\} = \mathcal{L}^R \mid \forall w \in \mathcal{L}$
- Гомоморфизм  $h(\mathcal{L})$   
Сопоставляет на каждый символ из начального языка - строку  
Пусть есть  $w \in \mathcal{L}$ .  $h(w) = h(a_1 \dots a_n) = h(a_1) + \dots + h(a_n)$   
 $h(\mathcal{L}) = \{h(w) \mid \forall w \in \mathcal{L}\}$
- Обратный гомоморфизм  $h^{-1}(\mathcal{L})$   
 $h^{-1}(h(\mathcal{L})) = \mathcal{L}$
- Симметрическая разность  $\mathcal{L}_1 \triangle \mathcal{L}_2 = (\mathcal{L}_1 \setminus \mathcal{L}_2) \cup (\mathcal{L}_2 \setminus \mathcal{L}_1)$
- Префикс (множество всех префиксов из регулярного языка)
- Субституция (Substitution)  
Сопоставляет на каждый символ из начального языка - язык  
Пусть есть  $w \in \mathcal{L}$ .  $f(w) = f(a_1 \dots a_n) = f(a_1) \cdot \dots \cdot f(a_n) = \mathcal{L}_{a_1} \cdot \dots \cdot \mathcal{L}_{a_n}$   
Тоест одной строке соответствует язык  $\iff$  1 и более строка.  
 $f(\mathcal{L}) = \bigcup f(w) \mid \forall w \in \mathcal{L}$

## 5 Pumping lemma

Если  $\mathcal{L}$  - регулярный язык, то существует  $n > 1$ , зависящее только от  $\mathcal{L}$ , такое, что  
 $\forall w \in \mathcal{L}, |w| > n \mid w = xyz$  так, что  
 $|y| > 0$   
 $|xy| < n$   
 $\forall k > 0 \mid xy^kz \in \mathcal{L}$

## 6 Regular expressions

Регулярное выражение - выражение описывающее регулярный язык.

Базис =  $Reg_0$ .

Если  $r_1$  и  $r_2$ , то следующие выражения также являются регулярными

- $r_1 r_2$  - конкатенация
- $r_1^*$  - звезда Клини (повторения от 0 до  $+\infty$ )
- $r_1^+$  - Позитивное дополнение..? (повторения от 1 до  $+\infty$ )
- $r_1^?$  - 0 или 1 повторение
- $\{expression\}\{n\}$  - повторение *expression* ровно  $n$  раз

- $\{expression\}\{min,\}$  - повторение  $expression$  хотя бы  $min$  раз
- $\{expression\}\{,max\}$  - повторение  $expression$  менее  $max$  раз
- $\{expression\}\{min,max\}$  - повторение  $expression$  более  $min$ , менее  $max$  раз
- $r_1 \cup r_2$  -  $r_1$  или  $r_2$   
 $r_1 + r_2$
- $[a_1...a_n]$  - ровно одно  $a_i$
- $()$  - для определения порядка применения операций

## 7 Deterministic finite automata (DFA)

Детерменированный конечный автомат (ДКА) - это кортеж  $(Q, \Sigma, \delta, q_0, F)$ , где

$Q$  - конечное множество состояний

$\Sigma$  - алфавит

$q_0 \in Q$  - начальное состояние

$F \subseteq Q$  множество принимающих состояний (терминалы)

$\delta: Q \times \Sigma \rightarrow Q$  - функция перехода

Автомат называется детерменированным так как из каждого состояния есть **ровно 1** переход для каждого символа

Впрочем, некоторые авторы говорят что ДКА это автомат у которого из каждого состояния есть **не более 1** перехода для каждого символа (если "нужен" переход, которого нет, то автомат останавливается)

(Формально, детерменированный автомат, это автомат который всегда находится в ровно одном состоянии)

## 8 Non-deterministic finite automata (NFA)

Недетерменированный конечный автомат (НКА) - это кортеж  $(Q, \Sigma, \delta, q_0, F)$ , где

$Q$  - конечное множество состояний

$\Sigma$  - алфавит

$q_0 \in Q$  - начальное состояние

$F \subseteq Q$  множество принимающих состояний (терминалы)

$\delta: Q \times (\Sigma \cup \{\epsilon\}) \rightarrow 2^Q$  - функция перехода

*Обратите внимание, изменилось описание только функции перехода*

Автомат называется недетерменированным, если для какого либо состояния для какого либо символа существует не ровно один переход, или если в автомате есть  $\epsilon$ -переход (переход не "достающий" символ из строки).

(Такой автомат может находится в нескольких состояниях одновременно)

## 9 Rabin-Scott powerset construction

Конструкция Рабина-Скотта (subset construction) преобразует НКА в эквивалентный ДКА.

*Дальше идет упрощенное описание, потому что на вики какая то фигня происходит, это кажется слишком жестко и ничерта не понятно*

Каждое состояние в результирующем ДКА - подмножество состояний исходного НКА.

Каждое состояние ДКА имеющее хотя бы одно терминальное состояние из НКА - тоже терминальное.

Входное состояние - объединение  $\epsilon$ -замыканий каждой входной точки.

Пусть есть состояние  $A$  из ДКА, и некоторый символ  $x$ , полученный из входной строки. Тогда

переход из  $A$  по  $x$  будет в состояние  $B$ , такое что  $B$  - множество всех состояний, достижимых из хотя бы одного состояния в  $A$ .

Более формально:

есть  $A = \{a_1 \dots a_n\}$  где  $a_i$  - состояние исходного НКА,

есть символ  $x$ , по которому мы вычисляем переход.

Из исходного НКА мы можем найти  $a_i \xrightarrow{x} \{b_{i1} \dots b_{im}\} = b_i$  - множество состояний достижимых в НКА из  $a_i$  при переходе по  $x$ .

Тогда  $B = \bigcup_{i=1}^n b_i$ .

И в результирующем ДКА  $A \xrightarrow{x} B$

## 10 Epsilon-NFA

$\varepsilon$ -НКА - НКА, допускающий  $\varepsilon$ -переходы (переходы без чтения символа). Используется для удобства построения из регулярных выражений. Для  $\varepsilon$ -НКА вводится понятие  $\varepsilon$ -замыкания:

$$\varepsilon\text{-closure}(P) = \{q \mid \exists q_0 \in P, q_0 \xrightarrow{* \varepsilon} q\}.$$

(Множество всех состояний, достижимых из данного состояния по  $\varepsilon$ -переходам)

Именно  $\varepsilon$ -замыкание начального состояния задаёт множество начальных состояний эквивалентного НКА без  $\varepsilon$ -переходов.

## 11 NFA construction from epsilon-NFA

*\*По сути - просто стягиваете  $\varepsilon$ -замыкания в одно состояние*

Для удаления  $\varepsilon$ -переходов строится эквивалентный НКА без них. Каждый переход по символу  $a$  из состояния  $q$  в  $\varepsilon$ -НКА заменяется переходом из каждого состояния  $p \in \varepsilon\text{-closure}(q)$  по символу  $a$  в состояния из  $\delta(p, a)$ . Формально:

$$\delta'(q, a) = \bigcup_{p \in \varepsilon\text{-closure}(q)} \delta(p, a),$$

при этом принимающие состояния

$$F' = \{q \mid \varepsilon\text{-closure}(q) \cap F \neq \emptyset\}.$$

Полученный  $\varepsilon$ -свободный НКА распознаёт тот же язык, что и исходный.

## 12 Kleene's theorem

*Kleene's theorem:* множество регулярных языков  $\iff$  множество регулярных выражений  $\iff$  множество ДКА

то есть для любого регулярного выражения существует ДКА (и НКА) с тем же языком, и наоборот, для любого ДКА есть регулярное выражение, задающее тот же язык.

Это означает, что конструкции между регулярными выражениями и автоматами обратимы, то есть регулярные языки = языки ДКА/НКА.

## 13 Kleene's algorithm

Алгоритм Клини строит регулярное выражение по ДКА через последовательное исключение состояний. Определяется регулярное выражение  $R_{ij}^{(k)}$  — все слова, переводящие автомат из состояния  $i$  в  $j$ , не используя промежуточные состояния с номерами более  $k$ . В базисе  $k = 0$  выражения соответствуют прямым переходам. Далее рекуррентно:

$$R_{ij}^{(k)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)} (R_{kk}^{(k-1)})^* R_{kj}^{(k-1)}.$$

В конце  $R_{s,t}^{(n)}$  между начальным и принимающими состояниями даёт искомое регулярное выражение.

## 14 Thompson's construction

Построение Томпсона превращает регулярное выражение в эквивалентный НКА (обычно с  $\varepsilon$ -переходами). Каждая операция регулярного выражения соответствует конструкции автомата:

- Для символа  $a$ : автомат из нового начального состояния в новое конечное по переходу  $a$ .
- Для  $r_1 + r_2$ : создаётся новое начальное состояние с  $\varepsilon$ -переходами в начала автоматов для  $r_1$  и  $r_2$ , а их концы соединяются с общим конечным состоянием через  $\varepsilon$ -переходы.
- Для конкатенации  $r_1 r_2$ : конец автомата  $r_1$  соединяется с началом  $r_2$  через  $\varepsilon$ -переход.
- Для  $r^*$ : от начала к концу добавляется  $\varepsilon$ -переход, а от конца к началу также  $\varepsilon$ -переход, обеспечивая циклическое повторение.

В результате получается НКА, принимающий тот же язык, что и исходное регулярное выражение.

## 15 Ordered arrangements

Упорядоченное размещение  $n$  элементов. Задаётся как упорядоченная пара

$$\langle \Sigma, s \rangle$$

$$s: [n] \rightarrow \Sigma$$

$$s(i) = x \in \Sigma$$

Короче говоря соответствие индекса позиции и того что в ней стоит.

Еще можно записать как кортеж, строку, последовательность

## 16 Permutations

Перестановка из  $n$  элементов - это упорядоченное размещение всех элементов, число которых равно

$$n! = 1 \cdot 2 \cdot \dots \cdot n.$$

## 17 k-permutations

По сути - упорядоченное размещение, т.к. из  $n$  элементов выбираем  $k$  и считаем их количество перестановок. Вообще это

$$A(n, k) = \frac{n!}{(n-k)!}.$$

## 18 Cyclic permutations

Циклическая перестановка  $n$  элементов -  $(n-1)!$ , т.к. представим перестановку  $P$ , так как цикл длиной  $n$ , то существует  $n$  циклически одинаковых перестановок. Отсюда

$$\frac{n!}{n} = (n-1)!$$

Аналогично для циклических  $k$ -перестановок

$$\frac{A(n, k)}{k} = \frac{n!}{k(n-k)!}$$

## 19 Unordered arrangements

Неупорядоченное размещение  $k$  элементов множества  $\Sigma$  это мультимножество  $S$  размера  $k$

$\Sigma = 0, 1, 2$

$k = 3$

все возможные  $S = \{\{0, 0, 0\}, \{0, 0, 1\}, \{0, 0, 2\}, \{0, 1, 1\}, \{0, 1, 2\}, \{0, 2, 2\}, \{1, 1, 1\}, \{1, 1, 2\}, \{1, 2, 2\}, \{2, 2, 2\}\}$

## 20 k-combinations

Какой то выбор  $k$  элементов из множества размера  $n$

$$C(n, k) = \binom{n}{m} = \frac{n!}{k!(n-k)!}$$

(Элементы различны, не повторяются)

## 21 Binomial theorem

Ну вот теорема: (Дальше то че? ну теорема и теорема)

$$(x + y)^n = \binom{n}{0} x^n y^0 + \binom{n}{1} x^{n-1} y^1 + \dots + \binom{n}{n} x^0 y^n$$

Ладно, минутка интересных фактов, которые по хорошему вы должны знать:

$$\binom{n}{m} = \binom{n-1}{m-1} + \binom{n-1}{m}$$

## 22 Multisets

Мультисет - обобщение множества, допускающее повторения элементов. Задается как упорядоченная пара

$\langle \Sigma, r \rangle$

$r: \Sigma \rightarrow \mathbb{N}$  ( $r(x)$  - количество объектов  $x$ ,  $x \in \Sigma$ )

## 23 Permutations of multisets

$$|P(\Sigma^*, n)| = \frac{n!}{r_1! \cdot \dots \cdot r_s!}$$

$n = |\Sigma^*|$  (всего элементов в мультимножестве),  $s = |\Sigma|$  (кол-во уникальных элементов в мультимножестве)

## 24 Combinations of infinite multisets

Сочетание с повторениями (или сочетание из бесконечного мультисета) из  $n$  типов элементов (каждого в неограниченном количестве) по  $k$  элементов даёт формулу:

$$\binom{n+k-1}{k}.$$

Доказывать можно по Stars&Bars



## 25 Multinomial theorem

$$(x_1 + x_2 + \dots + x_m)^n = \sum_{k_1 + k_2 + \dots + k_m = n} \frac{n!}{k_1! k_2! \dots k_m!} x_1^{k_1} x_2^{k_2} \dots x_m^{k_m}$$
$$\forall k_i \geq 0$$

где сумма берётся по неотрицательным целым  $(n_1, \dots, n_m)$ , сумма которых равна  $n$ . Коэффициенты  $\frac{n!}{k_1! \dots k_m!}$  называются мультиномиальными и равны числу способов разбить  $n$  объектов на группы размеров  $k_1, \dots, k_m$ .

## 26 Compositions

Слабая композиция числа  $n \geq 0$  на  $k$  частей это решение уравнения  $b_1 + \dots + b_k = n$  ( $b_i \geq 0$ )  
Их количество (по Stars&Bars)

$$\binom{n+k-1}{k-1}$$

Композиция числа  $n \geq 0$  на  $k$  частей это решение уравнения  $b_1 + \dots + b_k = n$  ( $b_i \geq 0$ )  
Их количество

$$\binom{n-1}{k-1}$$

кол-во делений  $n$  на произвольное кол-во частей ( $k = 1 \dots n$ )  $= 2^{n-1}$  (Опаньки, биномиальная теорема пригодились)

## 27 Set partitions

Партиция (разбиение) множества из  $n$  элементов на  $k$  блоков (неупорядоченных и непустых)  
Считается как число стирлинга второго рода

$$S(n, k) = S(n-1, k-1) + k \cdot S(n-1, k)$$

Если  $k = 1 \dots n$ , то вычисляется как число Белла.

$$B(n) = \sum_{k=0}^n S(n, k)$$

## 28 Stirling numbers of the second kind

Числа Стирлинга второго рода соответствуют количеству разбиений множества из  $n$  элементов на  $k$  непустых подмножеств. Их можно вычислить по формуле:

$$S(n, k) = S(n-1, k-1) + k \cdot S(n-1, k)$$
$$S(n, k) = 1 \Leftrightarrow n = k$$
$$S(n, 0) = 0$$
$$S(n, k) = 0 \Leftrightarrow n < k$$

Формула построена из следующих соображений: пусть нам нужно разделить множество из  $n$  элементов на  $k$  непустых частей. У нас есть два варианта:

- Разделить  $n-1$  элемент на  $k-1$  частей, а оставшийся элемент добавить в отдельную часть
- Разделить  $n-1$  на  $k$  частей, а оставшийся элемент добавить в одну из получившихся частей. Т.к. у нас получилось  $k$  частей, а оставшийся элемент можно добавить в любую из них  $\rightarrow$  получаем  $k$  вариантов (для каждого разбиения  $n-1$  на  $k$  частей)

## 29 Bell numbers

Число Белла  $B_n$  равно числу всех разбиений  $n$ -элементного множества.

$$B(n) = \sum_{k=0}^n S(n, k)$$

$$B(n) = \sum_{k=0}^{n-1} \binom{n-1}{k} B(k)$$

$$\sum_{n=0}^{\infty} B_n \frac{x^n}{n!} = \exp(e^x - 1)$$

Некоторые значения:  $B_0 = 1$ ,  $B_1 = 1$ ,  $B_2 = 2$ ,  $B_3 = 5$ , ...

## 30 Integer partitions

Целочисленное разбиение  $p(n)$  - число способов представить  $n$  как сумму неубывающих натуральных слагаемых (порядок **неважен** (В композициях важен, по тому и формулы разные)))  
Например,  $p(4) = 5$  для разбиений

$$4, 3 + 1, 2 + 2, 2 + 1 + 1, 1 + 1 + 1 + 1.$$

$$p(n, k) = p(n - 1, k - 1) + p(n - k, k)$$

$$p(0, 0) = 1$$

$$p(n, k) = 0 \Leftarrow (n \leq 0 \cup k \leq 0)$$

## 31 Principle of Inclusion-Exclusion

Для конечных множеств  $A_1, \dots, A_m$  мощность объединения даётся формулой:

$$\left| \bigcup_{i=1}^m A_i \right| = \sum_i |A_i| - \sum_{i < j} |A_i \cap A_j| + \sum_{i < j < k} |A_i \cap A_j \cap A_k| - \dots + (-1)^{m-1} |A_1 \cap \dots \cap A_m|.$$

Этот принцип позволяет учитывать пересечения при подсчёте, например, количество объектов, не обладающих ни одним из нежелательных свойств. В частности, число перестановок без фиксированных точек (derangements) можно получить:

$$!n = n! \sum_{k=0}^n \frac{(-1)^k}{k!}.$$

## 32 Recurrence relations

Рекуррентное (или рекурсивное) соотношение — это уравнение, задающее  $a_n$  через предыдущие члены. Линейное однородное соотношение с постоянными коэффициентами имеет вид:

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

Общее решение строится через корни характеристического многочлена

$$\lambda^k - c_1 \lambda^{k-1} - \dots - c_k = 0.$$

Если корни  $\lambda_1, \dots, \lambda_r$  различны, то  $a_n = \sum_i \alpha_i \lambda_i^n$ . Для кратных корней добавляются множители  $n$ ,  $n^2$  и т.д.

У неоднородных есть свободные члены (что то, что не зависит от последовательности (+3, +n))

Однородные:  $a_n = a_{n-1} + a_{n-2}$ ;  $a_n = 2a_{n-1}$

Неоднородные:  $a_n = a_{n-1} + n$

### 33 Solving recurrence relations using characteristic equations

Для решения однородных линейных рекурренций ищут корни характеристического уравнения

$$\lambda^k - c_1\lambda^{k-1} - \dots - c_k = 0.$$

Если, например, корень  $\lambda$  имеет кратность  $m$ , то вклад от него даётся  $\alpha n^{m-1}\lambda^n$ . Для неоднородных рекуррентных ищут сначала частное решение (например, подбирая вид, подобный неоднородной части), а затем добавляют общее решение однородного уравнения.

### 34 Generating functions

Порождающая функция последовательности  $(a_n)$  — формальный степенной ряд

$$A(x) = \sum_{n=0}^{\infty} a_n x^n.$$

Она кодирует последовательность и позволяет решить рекуррентные соотношения с помощью алгебраических манипуляций над рядами. Операции над порождающими функциями:

- Сумма рядов:  $A(x) + B(x) = \sum (a_n + b_n)x^n$ .
- Произведение рядов:  $A(x)B(x) = \sum_{n \geq 0} \left( \sum_{k=0}^n a_k b_{n-k} \right) x^n$  (свёртка коэффициентов).
- Стандартные ряды:  $\frac{1}{1-x} = \sum_{n=0}^{\infty} x^n$ ,  $\frac{1}{(1-x)^m} = \sum_{n=0}^{\infty} \binom{n+m-1}{m-1} x^n$ .

### 35 Solving linear recurrences using generating functions

Для линейного рекуррента составляют уравнение для  $A(x)$ , решают его и затем извлекают коэффициенты. Например, для Фибоначчи

$$a_n = a_{n-1} + a_{n-2}, \quad a_0 = 0, \quad a_1 = 1$$

получаем

$$A(x) = xA(x) + x^2A(x) + x,$$

откуда

$$A(x) = \frac{x}{1-x-x^2}$$

и извлекается формула

$$a_n = \frac{1}{\sqrt{5}} \left( \left( \frac{1+\sqrt{5}}{2} \right)^n - \left( \frac{1-\sqrt{5}}{2} \right)^n \right).$$

### 36 Solving combinatorial problems using generating functions

Для комбинаторных классов строят порождающие функции по правилам: правило суммы соответствует объединению классов (сумма ГФ), правило произведения — декартово произведение (произведение ГФ). Например, если имеется  $m$  типов букв, каждая может повторяться любое число раз, то полная ГФ равна

$$(1 + x + x^2 + \dots)^m = \frac{1}{(1-x)^m},$$

и коэффициент при  $x^n$  равен

$$\binom{n+m-1}{m-1}$$

(числу слов длины  $n$ ).

## 37 Operators and annihilators

Линейный разностный оператор  $E$  действует как  $E(a_n) = a_{n+1}$ . Оператор  $\Delta = E - 1$  даёт разность  $\Delta(a_n) = a_{n+1} - a_n$ . Аннигилятором называется такой оператор, который обращает последовательность в нулевую. Например, оператор  $E - 1$  аннигилирует константу (так как  $a_{n+1} - a_n = 0$ ), оператор  $E - 2$  аннигилирует геометрическую прогрессию  $2^n$  ( $2a_n - a_{n+1} = 0$ ). Идея: найти оператор, аннигилирующий неоднородную часть  $f(n)$ , и применить его к исходному уравнению, чтобы получить однородное соотношение.

## 38 Solving linear recurrences using annihilators

Если дано рекуррентное соотношение

$$a_n + c_1 a_{n-1} + \dots + c_k a_{n-k} = f(n),$$

ищут оператор  $P(E)$  такой, что  $P(E)f(n) = 0$ . Умножая исходное уравнение оператором  $P(E)$ , получают новое однородное соотношение

$$P(E)(E^k + \dots + c_k) = 0$$

для  $a_n$ , решают его, затем отбрасывают лишние решения, чтобы учесть начальные условия исходного уравнения.

## 39 Catalan numbers

Числа Каталана  $\{C_n\}$  можно определить, например, как количество правильных скобочных последовательностей длины  $2n$  или число способов разбить  $(n+2)$ -угольник на треугольники. Формула:

$$C_n = \frac{1}{n+1} \binom{2n}{n}.$$

Рекуррентно:  $C_0 = 1$ ,

$$C_{n+1} = \sum_{i=0}^n C_i C_{n-i}.$$

Генерирующая функция  $C(x) = \sum_{n \geq 0} C_n x^n$  удовлетворяет уравнению

$$C(x) = 1 + x C(x)^2,$$

что даёт замкнутую форму

$$C_n = \frac{1}{n+1} \binom{2n}{n}.$$

## 40 Generalized binomial theorem

Для любого действительного (или комплексного)  $r$  справедливо:

$$(1+x)^r = \sum_{k=0}^{\infty} \binom{r}{k} x^k, \quad \text{где } \binom{r}{k} = \frac{r(r-1)\dots(r-k+1)}{k!}.$$

Ряд бесконечен и сходится при  $|x| < 1$ . Для натурального  $r$  ряд конечен и совпадает с обычной биномной теоремой.

## 41 Gamma function

Гамма-функция  $\Gamma(x)$  продолжает факториал на вещественные и комплексные значения:

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt,$$

при этом  $\Gamma(n) = (n-1)!$  для натуральных  $n$ . Рекуррентно:  $\Gamma(x+1) = x\Gamma(x)$ . Гамма-функция играет роль обобщённого факториала во многих формулах.

## 42 Divide-and-Conquer algorithms analysis using recursion trees

Метод рекурсивных деревьев используется для оценки сложности рекурсивных алгоритмов вида

$$T(n) = aT\left(\frac{n}{b}\right) + f(n).$$

Строят дерево рекурсии: на  $i$ -м уровне  $a^i$  узлов, каждый параметром  $n/b^i$  и затратами  $f(n/b^i)$ . Суммарное время

$$T(n) = \sum_{i=0}^{\log_b n} a^i f\left(\frac{n}{b^i}\right).$$

Сравнивая вклад каждого уровня, можно определить асимптотику  $T(n)$ .

## 43 Master theorem

Рассматривается рекуррент

$$T(n) = aT\left(\frac{n}{b}\right) + f(n),$$

где  $a \geq 1$ ,  $b > 1$  и  $f(n)$  положительная функция. Тогда:

- Если  $f(n) = O(n^{\log_b a - \varepsilon})$  для некоторого  $\varepsilon > 0$ , то  $T(n) = \Theta(n^{\log_b a})$ .
- Если  $f(n) = \Theta(n^{\log_b a} \log^k n)$  для некоторого  $k \geq 0$ , то  $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$ .
- Если  $f(n) = \Omega(n^{\log_b a + \varepsilon})$  для некоторого  $\varepsilon > 0$  и при этом  $a f(n/b) \leq c f(n)$  для некоторого  $c < 1$  при достаточно больших  $n$ , то  $T(n) = \Theta(f(n))$ .

## 44 Akra–Bazzi method

Обобщение Мастер-теоремы для более сложных рекуррент:

$$T(x) = \sum_{i=1}^k a_i T(b_i x + g_i(x)) + f(x),$$

где  $a_i > 0$ ,  $0 < b_i < 1$ ,  $g_i(x) = O(x/\log^2 x)$ ,  $f(x)$  неотрицательна. Сначала решают уравнение

$$\sum_{i=1}^k a_i b_i^p = 1$$

относительно  $p > 0$ . Тогда оценка даётся формулой

$$T(x) = \Theta\left(x^p \left(1 + \int_1^x \frac{f(u)}{u^{p+1}} du\right)\right).$$

Этот метод позволяет получать асимптотику рекуррентных соотношений, где аргументы рекурсии уменьшаются неравномерно.