

ДЗ 3

Cosmo Chief

6 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	Regular expressions	5
8	Deterministic finite automata (DFA)	5
9	Non-deterministic finite automata (NFA)	5
10	Rabin–Scott powerset construction	5
11	Epsilon-NFA	6
12	NFA construction from epsilon-NFA	6
13	Kleene’s theorem	6
14	Kleene’s algorithm	6
15	Thompson’s construction	6
16	Ordered arrangements	7
17	Permutations and cyclic permutations	7
18	Unordered arrangements	7
19	Multisets	7
20	Combinations of infinite multisets	8
21	Multinomial theorem	8
22	Compositions	8
23	Set partitions and Stirling numbers of the second kind	8
24	Bell numbers	8

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

1 Formal languages

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

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

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
 $\mathcal{L}^k = \underbrace{\mathcal{L} \cdot \dots \cdot \mathcal{L}}_{k \text{ раз}}$
 $\mathcal{L}^0 = \{\epsilon\}, \epsilon = \text{пустое слово}$
- Звезда Клини
 $\mathcal{L}^* = \bigcup_{k=0}^{\infty} \mathcal{L}^k$

3 Regular languages

Множество регулярных языков задается так:

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

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. То есть $\mathcal{L}^+ \subseteq \mathcal{L}^*$
- 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 ϵ -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 $w \in \mathcal{L}$. $f(w) = f(a_1 \dots a_n) = f(a_1) \cdot \dots \cdot f(a_n) = \mathcal{L}_{a_1} \cdot \dots \cdot \mathcal{L}_{a_n}$
То есть 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
- $r_1|r_2$ - r_1 или r_2
- $[a_1...a_n]$ - ровно одно a_i
- $()$ - для определения порядка применения операций

7 Deterministic finite automata (DFA)

ДКА — это кортеж $(Q, \Sigma, \delta, q_0, F)$, где Q — конечное множество состояний, Σ — алфавит, $q_0 \in Q$ начальное состояние, $F \subseteq Q$ множество принимающих состояний, $\delta: Q \times \Sigma \rightarrow Q$ — функция перехода. Автомат в состоянии q по символу a переходит в состояние $\delta(q, a)$. Язык автомата — множество слов, на которых автомат после чтения остаётся в принимающем состоянии. ДКА распознают регулярные языки. Например, автомат с $Q = \{q_0, q_1\}$, $\Sigma = \{0, 1\}$, q_0 начальное, $F = \{q_1\}$ и переходом $\delta(q_0, 1) = q_1$, $\delta(q_1, 0) = q_1$, остальные переходы в себя, распознаёт все слова, содержащие хотя бы одну единицу.

8 Non-deterministic finite automata (NFA)

НКА — это кортеж $(Q, \Sigma, \delta, q_0, F)$, где $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$. Здесь по входному символу (или пустому переходу ε) может быть несколько возможных переходов или ни одного. Слово принимается, если существует путь из q_0 по символам слова (с учётом ε -переходов) в принимающее состояние. Язык НКА совпадает с регулярным языком. НКА удобен при построении автомата из регулярного выражения.

9 Rabin–Scott powerset construction

Конструкция Рабина–Скотта (subset construction) преобразует НКА в эквивалентный ДКА. Алгоритм: множество состояний ДКА соответствует подмножествам состояний НКА. Начальное состояние ДКА — ε -замыкание начального состояния НКА. Для каждого множества $P \subseteq Q$ и символа a вычисляется

$$\delta'(P, a) = \bigcup_{q \in P} \delta(q, a),$$

а затем берётся его ε -замыкание. Множество принимающих состояний нового автомата — все P , содержащие хотя бы одно принимающее состояние НКА. В результате получаем ДКА, распознающий тот же язык, что и исходный НКА.

10 Epsilon-NFA

ε -НКА — НКА, допускающий ε -переходы (переходы без чтения символа). Формально $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$. Используется для удобства построения из регулярных выражений. Для ε -НКА вводится понятие ε -замыкания:

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

Именно ε -замыкание начального состояния задаёт множество начальных состояний эквивалентного НКА без ε -переходов.

11 NFA construction from epsilon-NFA

Для удаления ε -переходов строится эквивалентный НКА без них. Каждый переход по символу a из состояния q в ε -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 ε -свободный НКА распознаёт тот же язык, что и исходный.

12 Kleene's theorem

- *Kleene's theorem*: Регулярные языки эквивалентны регулярным выражениям; 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 Клеене 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 ε -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 ε -переходами в начала автоматов для r_1 и r_2 , а их концы соединяются с общим конечным состоянием через ε -переходы.
- Для конкатенации $r_1 r_2$: конец автомата r_1 соединяется с началом r_2 через ε -переход.
- Для r^* : от начала к концу добавляется ε -переход, а от конца к началу также ε -переход, обеспечивая цикличное повторение.

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

15 Ordered arrangements

Упорядоченное размещение из n элементов по k (обозначается $A(n, k)$) — выборка без повторений, упорядоченная. Количество:

$$A(n, k) = n(n-1) \cdots (n-k+1) = \frac{n!}{(n-k)!}.$$

Например, $A(5, 3) = 5 \times 4 \times 3 = 60$. При $k = n$ это число перестановок $n!$.

16 Permutations and cyclic permutations

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

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

k -перестановка — это частный случай упорядоченного размещения, равен $A(n, k)$. Циклическая перестановка (размещение по кругу) из n элементов учитывает циклическую симметрию, и их число равно

$$(n-1)!,$$

так как можно зафиксировать один элемент и переставлять остальные $n-1$.

17 Unordered arrangements

Неупорядоченное размещение (сочетание) из n по k — выборка k элементов из n без учёта порядка. Обозначается $\binom{n}{k}$. Количество:

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

Свойства: $\binom{n}{0} = 1$, $\binom{n}{n} = 1$, $\binom{n}{k} = \binom{n}{n-k}$. Бином Ньютона:

$$(x+y)^n = \sum_{k=0}^n \binom{n}{k} x^k y^{n-k}.$$

18 Multisets

Мультисет — обобщение множества, допускающее повторения элементов. Мультисет задаётся так: у элементов типов $1, 2, \dots, m$ есть количества n_1, n_2, \dots, n_m , общее число элементов $n = n_1 + \dots + n_m$. Число перестановок такого мультисета:

$$\frac{n!}{n_1! n_2! \dots n_m!}.$$

19 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:

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

Это соответствует количеству неубывающих последовательностей длины k из n типов или представлению k в виде суммы n неотрицательных слагаемых.

20 Multinomial theorem

Обобщённая биномиальная теорема:

$$(x_1 + x_2 + \dots + x_m)^n = \sum_{n_1+n_2+\dots+n_m=n} \frac{n!}{n_1! n_2! \dots n_m!} x_1^{n_1} x_2^{n_2} \dots x_m^{n_m},$$

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

21 Compositions

Композиция целого n — упорядоченное разбиение $n = i_1 + i_2 + \dots + i_k$ на k положительных частей. Количество композиций n в неограниченном числе частей равно 2^{n-1} (между n единицами можно ставить либо разделитель, либо нет). Более детально: число композиций n на k частей равно $\binom{n-1}{k-1}$ (выбираем $k-1$ разбиений среди $n-1$ возможных).

22 Set partitions and Stirling numbers of the second kind

Партиция (разбиение) множества из n элементов на k блоков (неупорядоченных и непустых) считается числом Стирлинга второго рода $\left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\}$. Оно удовлетворяет рекурсии:

$$\left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\} = k \left\{ \begin{smallmatrix} n-1 \\ k \end{smallmatrix} \right\} + \left\{ \begin{smallmatrix} n-1 \\ k-1 \end{smallmatrix} \right\}, \quad \left\{ \begin{smallmatrix} n \\ 1 \end{smallmatrix} \right\} = 1, \quad \left\{ \begin{smallmatrix} n \\ n \end{smallmatrix} \right\} = 1.$$

Число Белла B_n — количество всех разбиений n -элементного множества (на любое число блоков):

$$B_n = \sum_{k=0}^n \left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\}.$$

23 Bell numbers

Число Белла B_n равно числу всех разбиений n -элементного множества. Генерирующая функция Белла:

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

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

24 Integer partitions

Целочисленное разбиение $p(n)$ — число способов представить n как сумму неубывающих натуральных слагаемых (порядок неважен). Например, $p(4) = 5$ для разбиений

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

Нет простых замкнутых формул, но есть рекуррентные соотношения и асимптотическая формула Харди–Рамануяна.

25 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:

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

26 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.

27 Solving recurrence relations using characteristic equations

Для решения однородных линейных рекуррентных соотношений ищут корни характеристического уравнения

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

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

28 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: $\frac{1}{1-x} = \sum_{n=0}^{\infty} x^n$, $\frac{1}{(1-x)^m} = \sum_{n=0}^{\infty} \binom{n+m-1}{m-1} x^n$.

29 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

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

откуда

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

и извлекается формула

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

30 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).

31 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.

32 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.

33 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

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

34 Generalized binomial theorem

Для любого действительного (или комплексного) r справедливо:

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

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

35 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.

36 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 $T(n)$.

37 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 $T(n) = \Theta(f(n))$.

38 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.