Contents

Лекція №1

1.1 Інтуітивні визначення

Алгоритм - чітка скінченна однозначно визначена послідовніть(порядок) дій/кроків, яка призводить до очікуваного результату.

Властивості алгоритмів:

- Коректність (алгоритм розв'язує те, що трбеа і правильно)
- Ефективність (скільки ресурсів буде витрачати алгоритм)

Позначення:

x - якась задача з параметрами.

$T(x)$ - часова складність (к-ть кроків які зробить алгоритм, щоб розв'язати певну задачу).

$S(x)$ - просторова складність (це к-ть додаткової пам'яті, щоб завершити алгоритм).

$n = |x|$ - розмір вхідних даних.

Складність буває:

1. У найгіршому випадку $T(n) = \max\limits_{x,\,|x|=n} T(x)$
2. У середньому $\overline{T}(n) = \sum\limits_{x:\,|x|=n} Pr(x)T(x)$

   $Pr(x)$ - ймовірність отримати складність x.
3. Майже завжди: майже $\forall x:\ T(x) = T_n$

1.2 Бітова та арифметична складність

Бітові операції

Архітектури працюють з регістрами = двійкові вектори фік. довжини n ($n = 16,\ 32,\ 64\ldots$)

Бітові (логічні) операції: $\vee$, $\wedge$, $\oplus$, $\gg$, $\ll$, $\ggg$, $\lll$.

Кожна операція реалізується окремим вентелем (gate) $\longrightarrow$ складність реалізації булевої функції = к-ть гейтів.

Рівнем вище можна вважати, що бітова операція над регістром атомарна.

Бітова складність = к-ть бітових операцій, яку виконує алгоритм.

Арифметичні операції

$x = (x_{n-1}, \ldots, x_1, x_0) = x_0 + 2x_1 + 4x_2 + \ldots + 2^{n-1}x_{n-1};\ x_i = \{0,\ 1\}$

Арифметичні операції: $+ \mod 2^n$, $- \mod 2^n$, $\cdot \mod 2^n$, $\div \mod 2^n$, порівняння.

Зазвичай бітова складність ариф. операцій $\sim n^{const}$

Адитивна складність: $+$, $-$, vs

Мультиплікативна складність: $\cdot$

Оцінювання ефективності:

1. Вибір множини базових операцій.
2. Підрахунок / оцінювання к-ті операцій.
3. Обмеження часу роботи у конкретному обчисл. середовищі.

Додавання у стовпчик.
Лема: $\forall i : C_i \in \{0, 1\}$
У найгіршому випадку $T(n) = 1 + 2(n-1) = 2n - 1$ - додавань цифр.
У найкращому випадку $T(n) = n \Rightarrow n \leqslant T(n) \leqslant 2n - 1$

1.3 Схема Горнера

$p(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_2 x^2 + ax + a_0$
Метод грубої сили: $1 + 2 + ... + n = \dfrac{n(n+1)}{2}$ множень чисел.
Горнер: $p(x) = (((a_n x + a_{n-1})x + a_{n-2})x + ... + a)x + a_0 \Rightarrow n$ множень чисел.
$x^a = x^{a_0 + 2a_1 + 2^2 a_2 + ... + 2^t a_t} = x^{a_0} \cdot (x^2)^{a_1} \cdot (x^{2^2})^{a_2} \cdot ... \cdot (x^{2^t})^{a_t}$
Найгірший випадок $2t$.
Середнє $t + \dfrac{1}{2}(t-1) + 1 = \dfrac{3}{2}t - \dfrac{1}{2}$; $T(t) = \Theta(t)$
$x^a = (...(((x^{a_t})^2 x^{a_{t-1}})^2 x^{a_{t-2}})^2 ...)x^0$

1.4 Числа Фібоначі

$f_0 = 0$, $f_1 = 1$, $f_{n+2} = f_{n+1} + f_n$, $n \geqslant 0$
Метод грубої сили
Алгоритм обчислення числа фібоначі: $A1(n) \longrightarrow f_n$

1. Якщо $n = 0$ - повернути 0.
2. Якщо $n = 1$ - повернути 1.
3. Інакше $n = 0$ - повернути $A1(n-1) + A1(n-2)$.

$T(n) = T(n-1) + T(n-2) + 3$, $n \geqslant 2$
$T(0) = 1$, $T(1) = 2 \Rightarrow T(n) \geqslant f_n$
Ідея "динамічного програмування"

1. $f[0] = 0$, $f[1] = 1$
2. Для всіх інших від 2 до n: $f[i] = f[i-1] + f[i-2]$
3. Повторюємо $f[n]$

$T(n) = n - 2$

Формула Біне

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

$T(n) = 2\log_2 n + 2\log_2 n + 1 = 4\log_2 n + 1$ - Множення ірраціональних чисел.

Матриці

$\mathbf{Q} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$

$\mathbf{Q^n} = \begin{bmatrix} f_{n+1} & f_n \\ f_n & f_{n-1} \end{bmatrix}$

$T(n) = 2\log_2 n$ - множень матриці.

$32\log_2 n$ - множень цілих чисел.

Лекція №2

2.1 Символи Ландау та нотація Кнута

$f, g : \mathbb{N} \to \mathbb{R}$

$f = o(g) \Leftrightarrow \lim\limits_{n\to\infty} \frac{f(n)}{g(n)} = 0$

$f = \mathsf{O}(g) \Leftrightarrow \exists C > 0, n_0 \in \mathbb{N}, \forall n \geqslant n_0 : \ |f(n)| \leqslant C \cdot |g(n)|$

$f = \omega(g) \Leftrightarrow \lim\limits_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ або $g = o(f)$

$f = \Omega(g) \Leftrightarrow \exists C > 0, n_0 \in \mathbb{N}, \forall n \geqslant n_0 : \ |f(n)| > C|g(n)|$ або $g = \mathsf{O}(f)$

$f = \Theta(g) \Leftrightarrow \exists C_1, C_2 > 0, n_0 \in \mathbb{N}, \forall n \geqslant n_0 : \ C_1|g(n)| \leqslant |f(n)| \leqslant C_2|g(n)|$

$f \sim g \Leftrightarrow \lim\limits_{n\to\infty} \frac{f(n)}{g(n)} = 1$ або $f(n) = g(n) + o(g(n))$

$\sum\limits_{k=1}^{n} \Theta(f(k)) \neq \Theta(f(1)) + \Theta(f(2)) + ... + \Theta(f(n))$

$\Theta(1)$ або $\mathsf{O}(n)$ - const.

$o(1)$ - нескінченно мала функція.

Твердження: $\lim\limits_{n\to\infty} \left|\frac{f(n)}{g(n)}\right| = C;\ 0 < C < \infty \Rightarrow f = \Theta(g)$

$$\left.\begin{matrix} f(n) = n\sin(n) \\ g(n) = n \end{matrix}\right\} \Rightarrow f = \mathsf{O}(g)$$

Твердження: $f = \mathsf{O}(g), \forall n \in \mathbb{N}, g(n) \neq 0 \Rightarrow \exists C > 0, \forall n \in \mathbb{N} : \ |f(n)| \leqslant C|g(n)|$

Наслідок: Ω та Θ аналогічно.

Твердження: $\Lambda \in \{\mathsf{O}, \Omega, \Theta\},\ f, g > 0,\ f = \Lambda$

Для $h : \mathbb{N} \to \mathbb{N}$ нехай $F(n) = \sum\limits_{k=1}^{h(n)} f(k)$, $G(n) = \sum\limits_{k=1}^{h(n)} g(k)$.

Тоді $F = \Lambda(G)$

2.2 Ієрархія функцій за швидкістю росту

Твердження: Нехай маємо поліном $p(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_2 x^2 + ax + a_0$; $a_n \neq 0$, $x \to \infty$.
Тоді:

1. $p(x) \in \Theta(x^n)$
   $p(x) \sim a_n x^n$
2. $p(x) = o(x^t)$, $\forall t > n$. Поліном домінується.
3. $p(x) = o(e^x)$. Поліном домінується.
4. $p(\ln, x) = o(x^t)$, $\forall t > 0$. Поліном домінується.

Теорема (про ієрархію функцій)
Нехай $0 < \varepsilon < 1$, $C > 1$, $n \in \mathbb{N}, n \to \infty$
Тоді:
$\varepsilon^n \prec C \prec \ln(\ln(n)) \prec \ln(n) \prec n^\varepsilon \prec n \prec n^C \prec C^n \prec n! \prec n^n \prec C^{C^n}$
$\prec$ - домінація.
Які можуть бути оцінки складності.

1. $T(n) = \Theta(f(n))$, $\left(= \mathsf{O}(f(n)),\ = \Omega(f(n))\right)$
2. $T(n) = f(n) + o(f(n))$ або $T(n) \sim f(n)$
3. $T(n) = f(n) + \Theta(g(n))$

Класи алгоритмів за часовою складністю:

1. Поліноміальні: $T(n) = \Theta(n^C)$
2. Експоненційні: $T(n) = \Theta(C^n)$ $\left(\Theta(C^{pol(n^y)})\right)$
3. Субекспоненційні: $n^2 \prec T(n) \prec C^n$

За просторовою складністю:

1. Константні: $S(n) = \Theta(1)$
2. Логарифмічні: $S(n) = \Theta(\ln(n))$
3. Поліноміальні: $S(n) = \Theta(n^C)$

Оцніка $f(n) = \mathsf{O(g)}$ точно, якщо $\exists$ послідовніть $n_{(k)}$, $k \in \mathbb{N}$, яка нескінченно зростає, що:
$\left.\begin{array}{l}\varphi = f(n_k)\\ \psi = g(n_k)\end{array}\right\} \Rightarrow \varphi(n_k) = \Omega(\psi(k))$ $\left(\Rightarrow \varphi(k) = \Theta(\psi(k))\right)$
Твердження $f = \mathsf{O}(g)$ - точна оцінка $\Leftrightarrow f(n) \neq o(g)$

Алгоритми факторизації

Задача: $N \in \mathbb{N}$. Знайти $d|N$, $1 < d < N$ або сказати, що N просте. (| - ділення)

1. Сито Ератосфена.
   Перебираємо d від 2 до $\dfrac{N}{2}$ та перевіряємо $N \vdots d$?
   $T(n) = \dfrac{N}{2} - 1 = \Theta(N)$
   $n = \log_2 N = T(n) = \Theta(2^n)$
2. Уточнення Фібоначі.
   Лема: $N = a \cdot b \Rightarrow \min\{a,\, b\} \leqslant \sqrt{N} \leqslant \max\{a,\, b\} \Rightarrow$ Перебираємо $d \leqslant \sqrt{N} \Rightarrow T(n) =$
   $= \Theta(\sqrt{N})\left(T(n) = \Theta(2^{n/2})\right)$
3. 1974: Алгоритм Лемана $T(N) = \Theta(\sqrt{N})$
   1975: Алгоритм Шенкса $T(N) = \Theta\sqrt[4]{N}$

Нотація Ленстри

$L_N[\alpha,\, C] = e^{(C+o(1))(\ln(N))^{\alpha}(\ln(\ln(n)))^{1-\alpha}}$, де $C > 0$, $0 \leqslant \alpha \leqslant 1$

Якщо $\alpha = 0 \Rightarrow$ поліноміальна.

Якщо $\alpha = 0 \Rightarrow$ експоненційна.

Якщо $\alpha_1 < \alpha_2 \Rightarrow L_N[\alpha_1,\, C_1] = o(L_N[\alpha_2,\, C_2])$

1981: Метод квадратичного сита $T(N) = L_N\left[\dfrac{1}{2},\, 1\right]$. Доведена оцінка.

1991-93: Метод сита числового поля $T(N) = L_N\left[\dfrac{1}{3},\, \left(\dfrac{64}{9}\right)^3\right]$

Лекція №3

3.1 Коректність алгоритмів

Алгоритм коректний:

1. Завершує роботу за скінченний час (завершуваність).
2. Результат роботи = розв'язок поствленої задачі.

Методи доведення коректності:

1. Інваріант алгоритму: певна величина / функція / об'єкт / твердження, які:
   (a) Перед початком роботи він є коректним.
   (b) Якщо перед довільним кроком є істинним $\Rightarrow$ після виконання також є істинним.
   (c) Після завершення алгоритму істинність інваріанту показує розв'язок задач.

   = інваріант циклу.
2. Напівінваріант алгоритму - функції, які монотонно спадають під час роботи алгоритму.

Приклади:

1. Сортування вставками.
2. Алгоритм Евкліда.
   $d = \gcd(a, b)$, $a \geqslant b$
   $\forall i: \ d = \gcd(r_{i-1}, r_i)$
   $f(i) = r_{i+1}$
3. (a) Поки $n > 3$ виконувати:
   (b) Якщо $n \vdots 2$, то $n = \dfrac{n}{2} + 1$
   (c) інакше $n = n + 1$
   $f(n) = n - (-1)^n$ - монотонно спадає

   (a) Поки $n > 1$ виконувати:
   (b) Якщо $n \vdots 2$, то $n = \dfrac{n}{2}$
   (c) інакше $n = 3n + 1$

3.2 Складність у найгіршому випадку

Алгоритм A, який за входом x обчислює $y = A(x)$.
Нехай $C_A^T(x)$ - часові витрати, $C_A^S(x)$ - витрати за пам'ятю.
Визначимо певну метрику $||x||$ (розмір входу) і шукаємо оцінки виду $C_A^*(x) \leqslant f_*(||x||)$
$x_n = \{x : ||x|| = n\}$
Складність A у найгіршому випадку $T_A(n) = \max\limits_{x \in X_n} C_A^T(x)$
Зауваження:

1. $\left.\begin{array}{c} C_A^T = \text{к-ть певних операцій} \\ C_A^S = \text{к-ть значень, які зберігаються} \end{array}\right\}$ цілочисельні.
2. Не враховуємо накладні витрати.
3. C_A^T залежить від витрат базових операцій.
4. $T_1(n) < T_2(n) \not\Rightarrow \forall x : C_1^T(x) < C_2^T(x)$
5. $A(x) = A_2(A_1(x)) \not\Rightarrow T(n) = T_1(n) + T_2(n)$
6. $T'(n)$, $T''(n)$ - складність за різними операціями $\Rightarrow T(n) \leqslant T'(n) + T''(n)$

Приклади:

1. Сортування вставками:
   Варіант 1: порівнюємо x_i з $x_{i-3}, x_{i-2}, ..., x_1$
   Варіант 2: порівнюємо x_i з $x_1, x_2, ..., x_{i-1}$
   Найгірший випадок: $x_1 > x_2 > ... > x_n$
   $T_1'(n) = T_1''(n) = 1 + 2 + ... + n - 1 = \dfrac{n(n-1)}{2}$
   $T_1(n) = n(n-1)$
2. К-ть порівнянь $= \begin{cases} k, \ k < i \\ k - 1, \ k = i \end{cases}$
   Найгірший випадок: $x_1 < x_2 < ... < x_n$
   К-ть обмінів: $i - k$
   Найгірший випадок: $x_1 > x_2 > ... > x_n$
   $T_2''(n) = \dfrac{n(n-1)}{2}$

Сумарна к-ть операцій: $\begin{cases} i, \ k < 1 \\ i - 1, \ k = i \end{cases}$
Найгірший випадок: $x_1 > x_2 > ... > x_n$
$T_2(n) = \sum\limits_{i=2}^{n} i = \dfrac{n(n+1)}{2} - 1 = \dfrac{n^2 + n - 2}{2}$
$\dfrac{T_1(n)}{T_2(n)} = \dfrac{n^2 - n}{\frac{n^2+n-2}{2}} \to 2$

3.3 Складність у середньому

Нехай на X_n задано розподіл $p_n(x)$.
Тоді складність алгоритму A у середньому: $\overline{T}_A(n) = \sum\limits_{x \in X_n} p_n(x) \cdot C_A^T(x)$

Зауваження:

1. $\overline{T}_A(n) \leqslant T_A(n)$
2. Якщо $\overline{T}'(n)$, $\overline{T}''(n)$ - складність за різними операціями, $\overline{T}(n) = \overline{T}'(n) + \overline{T}''(n)$
3. Складність у середньому не завжди адекватно відображає складність алгоритму.

Приклади:

1. Сортування вставками.
   $\overline{T}'(n)$ - сер. к-ть порівнянь.
   Нехай ξ_i - витрати на кроці i, $i = \overline{2,n} \Rightarrow \overline{T}'(n) = M(\xi_2 + \xi_3 + ... + \xi_n) = \sum\limits_{i=2}^{n} M(\xi_i)$

Нехай $H_{u,v}$ - подія у перестановці $(x_1, ..., x_n)$ серед елементів $x_1, ..., x_{v-1}$ рівно u елементів меньші за x_v.

1. Обираємо $x_1, ..., x_v$ - C_n^k варіантів.
2. x_v - елемент номер $u+1$ за зростанням.
   $x_1, ..., x_{v-1}$ - представлень як завгодно $(v-1)!$
3. $x_{v+1}, ..., x_n$ - $(n-v)! \Rightarrow$ к-ть $C_n^k(v-1)!(n-v)! = \dfrac{n!}{v} \Rightarrow Pr\{H_{u,v}\} = \dfrac{1}{v}$

$N_{k,i}$ - к-ть порівнянь, якщо x_i потрібно поставити на k.

$$N_{k,i} = \begin{cases} i-k+1, \ k > 1 \\ i-1, \ k = 1 \end{cases}$$

$$M\xi_i = \sum_{k=1}^{i} N_{k,i} \cdot Pr\{H_{k,i}\} = \frac{1}{i}\left((i-1) + \sum_{k=2}^{i}(i-k+1)\right) = \frac{1}{i}((i-1)+1+2+3+...+(i-1)) =$$

$$= \frac{1}{i}\left(\frac{i(i+1)}{2} - 1\right) = \frac{i}{2} + \frac{1}{2} - \frac{1}{i}$$

$$\overline{T}'(n) = \sum_{i=2}^{n}\left(\frac{i}{2} + \frac{1}{2} - \frac{1}{i}\right) = \frac{1}{2}\left(\frac{n(n+1)}{2} - 1\right) + \frac{n-1}{2} - \left(\frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n}\right)$$

Оскільки $H_n = \ln(n) + \Theta(1)$, то $\overline{T}'(n) = \dfrac{n^2}{4} + \Theta(n)$

Нехай η_i - к-ть обмінів на i-тому кроці $\Rightarrow \xi_i < \eta_i \leqslant \xi_i + 1 \Rightarrow \overline{T}''(n) = \dfrac{n^2}{4} + \Theta(n)$

Лекція №4

4.1 Нижні границі складності та оптимальні алгоритми

$f(n)$ - нижня границя складності для алгоритмів класу $\mathfrak{A}$, якщо $\forall A \in \mathfrak{A}$: $T_A(n) \geqslant f(n)$
Приклади:

1. Пошук min/max елементу у довільному масиві.
   $T(n) \geqslant n - 1$
   Якщо треба min & max, то
   $T(n) \geqslant \left\lceil \frac{2n}{2} \right\rceil - 2$
2. Піднесення до степеня множенням.
   $a \to a^m$
   Початок: a
   Середина $a^{m_1}, a^{m_2}, ..., a^{m_t}$
   $\max\{a^{m_1}, ..., a^{m_t}\}$ не більше ніж подвоється.
   Кінець: $a^m \Rightarrow$ к-ть кроків $\geqslant \left\lceil \log_2 m \right\rceil$
3. Сортування за рахунок порівнянь та обмінів.
   Може бути реалізовано за допомогою бінарного дерева.
   Складність = max висоти дерева
   листів $= n! \Rightarrow 2^{T(n)} \geqslant n! \Rightarrow T(n) \geqslant \log_2 n! \geqslant n \log_2 n - 2n$

$A \in \mathfrak{A}$ - оптимальний за складністю, якщо $T_A(n)$ - нижня границя складності $\forall A' \in \mathfrak{A}$
$f(n)$ - асимптотична нижня границя для $\mathfrak{A}$, якщо $\forall A \in \mathfrak{A} : T_A(n) = \Omega(f(n))$
Асимптотично оптимальний за складністю алгоритм. Це такий алгоритм з класу $\mathfrak{A}$ складність якого є асимптотичною нижньою границею для всіх інших алгоритмів.

4.2 Метод грубої сили

Прямий підхід до розв'язування задач зазвичай застосованийна на безпосередньому формулюванні задачі, її об'єктів та концепцій.
У комбінаторних задачах - метод вичерпного перебору.
Зауваження:

1. Застосовний до дуже широкого спектру задач.
2. Для багатьох задач дає цілком раціональні розв'язки.
3. Вартість розробки більш ефективного алгоритму може бути завелика, якщо задач небагато, а грубою силою вони ров'язуються за прийнятний час.
4. Навіть якщо М.Г.С(метод грубої сили) дає неефективний алгоритм у загальному випадку, він може бути ефективним для невеликих задач.

5. Може виступати еталоном складності.

Приклади:

1. Пошук min елементу у масиві.
2. Додавання у стовпчик - цілком раціонально.
   Множення у стовпчки $\Theta(n^2)$
   Але для $n \leqslant 512$ бітів краще МГС (множення у стовпчик).
3. Задача комівояжера.
   Є граф, шукаємо послідовніть вершин $V_0, V_1, ..., V_{n-1}$ $V_0 = V - n$. (V_{n-1}, V_n) - ребра.
   Лобовий метод $n!$
   Фіксуємо першу вершину - $(n-1)!$
   Фіксуємо напрямок обходу - $\dfrac{(n-1)!}{2}$

4.3 Алгоритм Карацуби

Метод декомпозиції (поділяй і володарюй).
X, Y - Z_n цифрові числа. (основа B)
$X \cdot Y$ у стовпчик: $T_1(2n) = 4n^2 + \Theta(n)$
$X = X_1 \cdot B^n + X_0$, $Y = Y_1 \cdot B^n + Y_0$
$X \cdot Y = (X_1 \cdot B^n + X_0) \cdot (Y_1 \cdot B^n + Y_0) = X_1Y_1 \cdot B^{2n} + (X_1Y_0 + X_0Y_1)B^n + X_0Y_0$
$T_2(2n) = 4T_1(n) + \Theta(n) = 4n^2 + \Theta(n)$
$(X_1 + X_0)(Y_1 + Y_0) = X_0Y_0 + X_1Y_1 + (X_1Y_0 + X_0Y_1)$
$X \cdot Y = X_1Y_1B^{2n} + ((X_1 + X_0)(Y_1 + Y_0) - X_0Y_0 - X_1Y_1)B^n + X_0Y_0$
$T_3(2n) = 3T_1(n) + \Theta(n) = 3n^2 + \Theta(n)$

$$T_k(n) = \begin{cases} \Theta(1),\ n = 1 \\ 3T\left(\dfrac{n}{2}\right) + \Theta(n),\ n > 1 \end{cases} \Rightarrow T_k = \Theta\left(n^{\log_2 3}\right)$$

Зауваження.

$$T_k(n) = T_k\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T_k\left(\left\lceil \frac{n}{2} \right\rceil\right) + T_k\left(\left\lceil \frac{n}{2} \right\rceil + 1\right) + \Theta(n)$$

Лекція №5

5.1 Метод декомпозиції

1. Вхідна задача розбивається на підзадачі, розмір яких у рази меньший.
2. Підзадачі розв'язуються рекурсивно, для маленьких підзадач можна використати інший алгоритм.
3. Розв'язок вхідної задачі збирається з розв'язків підзадач (за необхідності).

$$\begin{cases} \varphi(n), n \leqslant n_0 \\ a \cdot T\left(\frac{n}{b} + f(n)\right), n > n_0 \end{cases}$$

n_0 - розмір атомарних задач
$\varphi(n)$ - складность розв'язку атомарних задач
Неатомарні задачі розбиваються на a підзадач розміру $\frac{n}{b}$
$f(n)$ - складність збірки.

Приклади:

1. Бінарний пошук.
   Вхід: відсортований масив A.
   Вихід: Номер a у A або помилка.
   $$T(n) = \begin{cases} \Theta(1), n = 1 \\ T\left(\frac{n}{2}\right) + \Theta(1), n > 1 \end{cases}$$
2. Пошук max підмасиву.
   Вхід: довільний масив A.
   Вихід: $i_{i,j} : A[i] + ...A[j] \to \max$
   Груба сила: C_n^2 підзадач, $T(n) = \Theta(n^2)$
   Декомпозиція:
   $\max A[i,j] = \max[i,k] + \max A[k+1,j]$; $i = 1...k$; $j = k+1...n$
   Складність: $\Theta(n)$
   $$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$$

5.2 Розв'язання рекурентних співвідношень методом підстановки

1. Метод підстановки.
   - інтуїтивно припускаємо який розв'язок і доводимо, що розв'язок саме такий.

Приклади:

1. $T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n$, $T(1) = 1$
   Припустимо: $T(n) = O(n \log_2 n) \Rightarrow \exists c > 0.\ T(n) \leqslant cn \log_2 n$
   $T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n \leqslant 2c\left\lfloor \frac{n}{2} \right\rfloor \log \left\lfloor \frac{n}{2} \right\rfloor + n \leqslant 2c\frac{n}{2}(\log_2 n - 1) + n = cn \log_2 n + n(1-c) \leqslant$
   $\leqslant cn \log_2 n$. При $c \geqslant 1$
   Але $T(1) = 1$, $n \log_2 n = 0 \Rightarrow n_0 = 2$
   $T(2) = 4 \leqslant c2 \log_2 n$
   $T(3) = 5 \leqslant c3 \log_2 n \Rightarrow c \geqslant 2$
2. $T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lceil \frac{n}{2} \right\rceil\right) + 1$
   $T(n) = O(n) \Rightarrow \exists c > 0 :\ T(n) \leqslant cn$
   $T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lceil \frac{n}{2} \right\rceil\right) + 1 \leqslant c\left(\left\lfloor \frac{n}{2} \right\rfloor + \left\lceil \frac{n}{2} \right\rceil\right) + 1 \leqslant cn + 1$
   $T(n) \leqslant cn - d$, $d > 0 \Rightarrow T(n) \leqslant cn + 1 - 2d \leqslant cn - d$, для $d \geqslant 1$
3. $T(n) = 2T(\sqrt{n}) + \log_2 n$
   Заміна $m = \log_2 n$, $S(m) = T(2^m)$
   $T(2^m) = 2T(2^{m/2}) + m$
   $S(m) = 2S\left(\frac{m}{2}\right) + m$
   $S(m) = O(m \log_2 m)$
   $T(n) = S(\log_2 n) = O(\log_2 n \cdot \log_2(\log_2 n))$

5.3 Оцінка дерева рекурсії

Приклади:

(див. перший малюнок дерева в лекції №05-5)

1. $T(n) = 3T\left(\frac{n}{2}\right) + \Theta(n) = 3T\left(\frac{n}{2}\right) + cn$

Рівень $k:\ 3^k$ задач.

Складність $\frac{cn}{2^k}$

Ост. рівень: $n^{\log_2 3}$ задач складності $T(1) = \Theta(1) \Rightarrow T(n) = \sum_{k=0}^{n-1} 3^k \cdot \frac{cn}{2^k} + \Theta(n^{\log_2 3}) =$

$= \left| t = \log_2 n \right| = \frac{cn(\frac{3}{2})^t - 1}{\frac{3}{2} - 1} + \Theta(n^{\log_2 3}) = \left| \frac{3^{\log_2 n}}{2^{\log_2 n}} = \frac{n^{\log_2 3}}{n} \right| = 2cn\frac{n^{\log_2 3}}{n} - 2cn + \Theta(n^{\log_2 3}) =$

$= \Theta(n^{\log_2 3})$

(див. другий малюнок дерева в лекції №05-5)

2. $T(n) = 3T\left(\frac{n}{2}\right) + \Theta(n^2) = 3T\left(\frac{n}{2}\right) + cn^2$ ("поганий"Карацуба)

Рівень $k:\ 3^k$ задач.

Складність $\frac{cn^2}{4^k}$

Ост. рівень: $n^{\log_2 3}$ задач складності $T(1) = \Theta(1) \Rightarrow T(n) = \sum_{k=0}^{n-1} 3^k \cdot \frac{cn^2}{4^k} + \Theta(n^{\log_2 3}) <$

$< cn^2 \sum_{k=0}^{\infty} \left(\frac{3}{4}\right)^k + \Theta(n^{\log_2 3}) = 4cn^2 + \Theta(n^{\log_2 3}) = O(n^2)$

Але: $T(n) \geqslant cn^2 \Rightarrow T(n) = \Theta(n^2)$

(див. третій малюнок дерева в лекції №05-5)

3. $T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + \Theta(n)$

Кожен рівень:

Складність: $\leqslant cn$

max висота $= \log_{3/2} n$

Тому $T(n) \leqslant cn \log_{3/2} n$

$T(n) = O(n \log_2 n)$

Лекція №6

6.1 Основна теорема методу декомпозиції

Нехай $T(n) = \begin{cases} \Theta(1), n \leqslant n_0 \\ aT\left(\frac{n}{b}\right) + f(n), n > n_0 \end{cases}$

$a \geqslant$, $b > 1$, $f(n) \geqslant 0$, $\frac{n}{b}$ - ціле число.

1. Якщо $\exists \varepsilon > 0 : f(n) = O(n^{\log_b n - \varepsilon})$, то $T(n) = \Theta(n^{\log_b a})$
2. Якщо $f(n) = \Theta(n^{\log_b a})$, то $T(n) = \Theta(n^{\log_b a} \log n)$
3. Якщо $\exists \varepsilon > 0 : \ f(n) = \Omega(n^{\log_b a + \varepsilon})$

$\exists 0 < c < 1 : \ af\left(\frac{n}{b}\right) \leqslant cf(n) \Rightarrow T(n) = \Theta(f(n))$ - Умова регулярності.

Приклади:

1. Алгоритм Карацуби. $T(n) = 3T\left(\frac{n}{2}\right) + \Theta(n)$
   $a = 3$, $b = 2$, $f(n) = n$, $n^{\log_b a} = n^{\log_2 3} \approx n^{0.58}$
   $\varepsilon = 0.1$, $n = O(n^{0.580-\varepsilon})$
   $\Rightarrow$ 1й випадок: $T(n) = \Theta(n^{\log_2 3})$
2. Бінарний пошук. $T(n) = T\left(\frac{n}{2}\right)$
   $a = 1$, $b = 2$, $f(n) = n^0 = 1$, $n^{\log_b a} = n^{\log_2 1} = n^0$
   $\Rightarrow$ 2й випадок: $T(n) = \Theta(n^0 \log n) = \Theta(\log n)$
3. Пошук максимального підмасиву. $T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$
   $a = 2$, $b = 2$, $f(n) = n$, $n^{\log_b a} = n^{\log_2 2} = n$
   $\Rightarrow$ 2й випадок: $T(n) = \Theta(n \log n)$
4. $T(n) = 3T\left(\frac{n}{4}\right) + n \log_2 n$
   $a = 3$, $b = 4$, $f(n) = n \log_2 n$, $n^{\log_b a} = n^{\log_4 3} \approx n^{0.79}$
   $\forall 0 < \varepsilon < 0.2 : \ f(n) = \Omega(n^{0.79+\varepsilon})$
   $3f\left(\frac{n}{4}\right) = 3\frac{n}{4} \log_2 \frac{n}{4} \leqslant \frac{3n}{4} \log_2 n$ для $c = \frac{3}{4}$
   $\Rightarrow$ 3й випадок: $T(n) = \Theta(n \log_2 n)$

5. $T(n) = 2T\left(\frac{n}{2}\right) + n\log_2 n$

   $a = 2,\ b = 2,\ f(n) = n\log_2 n,\ n^{\log_b a} = n^1$

   $f(n) = \Omega(n)$

   Але $\forall \varepsilon > 0$: $f(n) = o(n^{1+\varepsilon})$

   $\Rightarrow$ теорема не застосовна.

Лема П: $c > 0,\ g(n) = 1 + c + c^2 + ... + c^n$

Тоді $g(n) = \begin{cases} \Theta(1), c < 1 \\ n + 1 = \Theta(n), c = 1 \\ \Theta(c^{n+1}), c > 1 \end{cases}$

$c < 1:\ 1 \leqslant g(n) < \sum\limits_{k=0}^{\infty} c^k = \frac{1}{1-c} = const$

$c = 1:\ g(n) = n + 1$

$c > 1:\ g(n) = \frac{c^{n+1} - 1}{c - 1} = \Theta(c^{n+1})$

6.2 Доведення основної теореми

Доведення основної теореми: $n = b^t,\ n_0 = 1$

(див. малюнок дерева в лекції №06-1)

$t + 1$ рівень $t = \log_b n$

k-й рівень a^k підзадач.

Складність підзадачі $f\left(\frac{n}{b^k}\right)$ або $\Theta(1)$

Складність останнього рівня $a^t\Theta(1) = a^{\log_b n}\Theta(1) = (b^{\log_b a})^{\log_b n}\Theta(1)$

$\Theta(1) = \Theta(n^{\log_b a})$

$T(n) = \Theta(n^{\log_b a}) + \sum\limits_{k=0}^{t-1} a^k f\left(\frac{n}{b^k}\right)$. Позначимо $W(n) = \sum\limits_{k=0}^{t-1} a^k f\left(\frac{n}{b^k}\right)$

1. $f(n) = O(n^{\log_b a - \varepsilon})$

   $\Rightarrow W(n) = O\left(\sum\limits_{k=0}^{t-1} a^k \left(\frac{n}{b^k}\right)^{\log_b a - \varepsilon}\right) \Rightarrow$

   $\Rightarrow n^{\log_b a - \varepsilon} \sum\limits_{k=0}^{t-1} \left(\frac{a}{b^{\log_b a - \varepsilon}}\right)^k = n^{\log_b a - \varepsilon} \sum\limits_{k=0}^{t-1} (b^\varepsilon)^k = n^{\log_b a - \varepsilon}\Theta(b^{\varepsilon t}) = \Theta(n^{\log_b a - \varepsilon} n^\varepsilon) = \Theta(n^{\log_b a}) \Rightarrow$

   $\Rightarrow W(n) = O(n^{\log_b a})$

   $T(n) = \Theta + O = \Theta(n^{\log_b a})$

2. $f(n) = \Theta(n^{\log_b a})$

   $\Rightarrow W(n) = O\left(\sum\limits_{k=0}^{t-1} a^k \left(\frac{n}{b^k}\right)^{\log_b a}\right) \Rightarrow$

   $\Rightarrow n^{\log_b a} \sum\limits_{k=0}^{t-1} \left(\frac{a}{b^{\log_b a}}\right)^k = n^{\log_b a} \cdot t \Rightarrow$

$W(n) = \Theta(n^{\log_b a} \cdot \log n)$

$T(n) = \Theta(n^{\log_b a}) + \Theta(n^{\log_b a} \log n) = \Theta(n^{\log_b a} \log n)$

3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$

   $W(n) \geqslant f(n) \Rightarrow W(n) = \Omega(f(n))$

   $a^k f\left(\frac{n}{b^k}\right) \leqslant c^k f(n),\ c < 1$

   $W(n) \leqslant f(n) \sum_{k=0}^{t-1} c^k = f(n) \cdot \Theta(1) \Rightarrow W(n) = O(f(n)) \Rightarrow W(n) = \Theta(f(n))$

   $T(n) = \Theta(n^{\log_b a}) + \Theta(f(n)) = \Theta(f(n))$

Доведення основної теореми: довільне n

1. $T(n) = aT\left(\left\lfloor \frac{n}{b} \right\rfloor\right) + f(n)$
2. $T(n) = aT\left(\left\lceil \frac{n}{b} \right\rceil\right) + f(n)$

$x \geqslant \lfloor x \rfloor > x + 1$

$x \leqslant \lceil x \rceil < x + 1$

Нехай n_k - розмір задачі на рівні k.

(1) $n_k = \begin{cases} n, k = 0 \\ \left\lfloor \frac{n_{k-1}}{b} \right\rfloor, k \geqslant 1 \end{cases} \Rightarrow n_k \leqslant \frac{n}{b^k}$

Якщо $t = \lfloor \log_b n \rfloor$, то $n_t \leqslant \frac{n}{b^{\lfloor \log_b n \rfloor}} < \frac{n}{b^{\log_b n - 1}} = \frac{n}{n/b} = b$

$\Rightarrow t + 1$ - рівень у дереві рекурсії.

Складність останнього рівня $a^{\log const} \leqslant a^t \Theta(1) = a^{\lfloor \log_b n \rfloor} \Theta(1) \leqslant a^{\log const} \Rightarrow \Theta(n^{\log a})$

$T(n) = \Theta(n^{\log_b a}) + \sum_{k=0}^{t-1} a^k f(n_k),\ t = \lfloor \log_b n \rfloor$

(2) $n_k = \begin{cases} n, k = 0 \\ \left\lceil \frac{n_{k-1}}{b} \right\rceil, k \leqslant 1 \end{cases}$

$n_0 = n$

$n_1 = \left\lceil \frac{n}{b} \right\rceil \leqslant \frac{n_0}{b} + 1$

$n_2 = \left\lceil \frac{n_1}{b} \right\rceil \leqslant \frac{n}{b} + 1 + \frac{1}{b}$

$n_k \leqslant \frac{n}{b^k} + 1 + \frac{1}{b} + ... + \frac{1}{b^k} < \frac{n}{b^k} + \frac{1}{1 - \frac{1}{b}} = \frac{n}{b^k} + \frac{b}{b-1}$

Для $t = \lfloor \log_b n \rfloor$

$n_t < \frac{n}{b^k} + \frac{b}{b-1} < const$

Зауваження та доповнення

1. Якщо рекурента виду $T(n) \leqslant ... \Rightarrow T(n) = O(...)$
   $T(n) \geqslant ... \Rightarrow T(n) = \Omega(...)$
2. Спрощений вигляд основної теореми:
   $$T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^d) \Rightarrow \begin{cases} \Theta(n^d), d > \log_b a \\ \Theta(n^d \log n), d = \log_b a \\ \Theta(n^{\log_b a}), d < \log_b a \end{cases}$$
3. З умови регулярності випливає, що $f(n) = \Omega(...)$
   Дадаткові випадки:
   (a) $f(n) = \Theta(n^{\log_b a} \log^m n)$(для $m > -1$) $\Rightarrow T(n) = \Theta(n^{\log_b a} \log^{m+1} n)$
   (b) $f(n) = \Theta\left(\frac{n^{\log_b a}}{\log n}\right) \Rightarrow T(n) = \Theta(n^{\log_b a} \log(\log n))$
   (c) $f(n) = \Theta\left(\frac{n^{\log_b a}}{\log^m n}\right)$(для $m > 1$) $\Rightarrow T(n) = \Theta(n^{\log_b a})$
4. Існують узагальнення основної теореми:
   (a) теорема Аккра-Баззі
   $$T(n) = \sum_i a_i T\left(\frac{a}{b_i} + b_i(n)\right) + f(n)$$

Лекція №7

7.1 Форми представлення поліномів

Поліноми:

1. Канонічне представлення: $p(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_i x + a_0$, $a_i \in \mathbb{F}(R)$, $a_n \neq 0 \Rightarrow$
   $\Rightarrow deg(p) = n$
   $a_n = 1$: унітарний (нормований) поліном.
   $p(x) \leftrightarrow (a_0, a_i, ..., a_{n-1}, a_n)$
   Операції:

   (a) Додавання: $(n+1)$ додавання коеф. $= \Theta(n)$

   (b) Множення полінома на скаляр: $\alpha \cdot p(x) \leftrightarrow (\alpha a_0, \alpha a_1, ..., \alpha a_n) \Rightarrow$
   $\Rightarrow (n+1)$ множення $= \Theta(n)$

   (c) Множення двох поліномів: $p(x) \cdot q(x)$, $p(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_i x + a_0$,
   $q(x) = b_n x^n + b_{n-1}x^{n-1} + ... + b_i x + b_0 \Rightarrow p(x) \cdot q(x) = c_0 + c_1 x + c_2 x^2 + ... + c_{2n}x^{2n}$
   $C_k = \sum_{i=0}^{k} a_i b_{x-i}$ - $\leqslant k+1$ множення.
   Загалом буде $\Theta(n^2)$ множень

   (d) Обчислення значення у точці.
   Схема Горнера $\Rightarrow n$ множень $= \Theta(n)$

2. Представлення поліному через корені:
   $p(x) = \beta(x - z_1)(x - z_2)...(x - z_n)$
   $p(x) \leftrightarrow (\beta_1(z_1, z_2, ..., z_n))$
   Операції:

   (a) Додавання - ?

   (b) Множення на скаляр: $\alpha \cdot p(x) \leftrightarrow (\alpha\beta, (z_1, ..., z_n)) \Rightarrow 1$ множення.

   (c) Множення поліномів: $p(x) = \beta_1(x - z_1)...(x - z_n)$, $q(x) = \beta_2(x - t_1)...(x - z_n)$
   $p(x) \cdot q(x) = \beta_1\beta_2(x - z_1)...(x - z_n) \cdot (x - t_1)...(x - t_n) \Rightarrow 1$ множення + об'єднання мультимножин коренів.

   (d) Обчислення значення у точці: $p(x_0) = \beta(x_0 - z_1)...(x_0 - z_n) \Rightarrow n$ множень.

Проблема: не знайдено ефект. алгоритмів точного знаходження коренів поліному.

Перехід до канонічної форми:

$(x - z_1)$ $\quad -z_1 = a_0,\ 1 = a_1$

$(x-z_1)(x-z_2) = x^2+(-z_1-z_2)x+z_1z_2$ $\quad z_1z_2 = a_0,\ -z_1-z_2 = a_1,\ 1 = a_2 \Rightarrow +1$ множення.

$(x - z_1)(x - z_2)(x - z_3)$ $\quad \Rightarrow +2$ множення.

$(x - z_1)(x - z_2)...(x - z_n)$ $\quad \Rightarrow +$ (n+1) множення.

$\alpha(x - z_1)(x - z_2)...(x - z_n)$ $\quad \Rightarrow$ n множень.

$\Rightarrow 1 + 2 + ... + n = \dfrac{n(n+1)}{2}$ множень $= \Theta(n^2)$

7.2 Інтерполяційна формула Лагранжа

3. Представлення поліному через значення у точках.
   Наслідок з основної теореми алгебри: поліном степеня n однозначно відновлюється за значенням у $n+1$ різних точках.
   $p(x) \leftrightarrow \{(x_i, y_i)\}, y_i = p(x_i), i = \overline{0,m}, m \geqslant n$
   Перехід від канонічного до даного: $(m+1)$ разів схема Горнера $\Rightarrow \Theta(mn)$
   Операції:

   (a) Додавання: m+1 додавань $(y_i + y'_i)$

   (b) Множення на скаляр: m+1 множень $\alpha \cdot y_i$

   (c) Множення двох поліномів: m+1 множень $(y_i \cdot y'_i)$

   Але:

   (a) Поліноми задаються на однакових $\{x_i\}$

   (b) $m \geqslant 2n$

4. Обчислення значення поліному в точці - ?
5. Перехід до канонічного представлення - ?

Інтерполяційна формула Лагранжа:

$\{(x_i, y_i)\}, i = \overline{0,m}$

$$p(x) = y_0\frac{(x-x_1)(x-x_2)...(x-x_m)}{(x_0-x_1)(x_0-x_2)...(x_0-x_m)} + y_1\frac{(x-x_0)(x-x_2)...(x-x_m)}{(x_1-x_0)(x_1-x_2)...(x_1-x_m)} +$$
$$+ y_2\frac{(x-x_0)(x-x_1)(x-x_3)...(x-x_m)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)...(x_2-x_m)} + ... + y_m\frac{(x-x_0)(x-x_1)...(x-x_{m-1})}{(x_m-x_0)(x_m-x_1)...(x_m-x_{m-1})}$$

Перехід до канонічної форми:

1. Обчислення $\{\lambda_k\}$, $\lambda = \dfrac{y_k}{(x_k-x_0)(x_k-x_1)...(x_k-x_m)}$ (немає $x_k - x_k$) $\Rightarrow$ m множень на один коеф. $\Rightarrow m(m+1)$ множень усього.
2. Обчислення у канонічній формі $\{\lambda_k(k)\}$: $\lambda_k(x-x_0)(x-x_1)...(x-x_m) \Rightarrow \dfrac{m(m+1)}{2}$ множень на один поліном. Усього $(m+1)$ поліном $\Rightarrow \Theta(m^3)$

3. $p(x) = l_0 + l_1(x) + ... + l_m(x)$

Значення поліному у точці:
$p(x) = \lambda_0(x - x_1)(x - x_2)...(x - x_m) + \lambda_1(x - x_0)(x - x_2)...(x - x_m) + ... + \lambda_m(x - x_0)...(x - x_{m-1})$

1. $\{\lambda_k\}$ - передобчислення $\Rightarrow \Theta(m^2)$ операцій, $\Theta(m)$ пам'яті.
2. Підстановка значення: m множень у кожному доданку $\Rightarrow m(m+1)$ множень загалом $= \Theta(m^2)$
   Додавання нової точки у опис поліному:
   (a) Обчислення значення $p()$
   (b) Переобчислення усіх $\{\lambda_k\}$

7.3 Інтерполяційна формула Ньютона

$p(x) \to \{(x_i, y_i)\}$
$p(x) = d_0 + d_1(x - x_0) + d_2(x - x_0)(x - x_1) + ... + d_m(x - x_0)(x - x_1)...(x - x_{m-1})$
Пояснення: $d_0 = y_0$, $d_1(x - x_0) = y_1 - y_0$, ...
$d_k = d_k(x_0, x_1, ..., x_k)$ - пронормовані Δ^k від $x_0, x_1, ...$

$$d_k = \sum_{t=0}^{k} \frac{y_t}{\prod\limits_{i \neq t}^{k}(x_t - x_i)}$$

$$d_0 = y_0$$

$$d_1 = \frac{y_0}{(x_0 - x_1)} + \frac{y_1}{(x_1 - x_0)}$$

$$d_2 = \frac{y_0}{(x_0 - x_1)(x_0 - x_2)} + \frac{y_1}{(x_1 - x_0)(x_1 - x_2)} + \frac{y_2}{(x_2 - x_0)(x_2 - x_1)}$$

1. Обчислення коеф. $\{d_k\}$
   Ідея: якщо зберігати усі доданки ($\Theta(m)$ пам'яті), то d_k обчислюється за k множень (множення доданків на нові коеф.) $+k$ множень (обчислення нового доданку) $\Rightarrow$
   $\Rightarrow 2k$ множень. $\Rightarrow$ разом буде $2(1 + 2 + ... + m) = m(m + 1)$ операцій.
2. Перехід до канонічної форми:
   $p(x) = d_0 + d_1(x - x_0) + d_2(x - x_0)(x - x_1) + ... + d_m(x - x_0)(x - x_{m-1})$
   $d_0 = 0$ - 0 операцій
   $d_1(x - x_0)$ - 1 операція
   $d_2(x - x_0)(x - x_1)$ - 2 операції
   $d_3(x - x_0)(x - x_1)(x - x_2)$ - 2 операції $x(x - x_2) + 2$ операції $xd_3 = 4$ операції
   $d_m(x - x_0)(x - x_{m-1})$ - $2(m - 1)$ операцій.
   $\Rightarrow \Theta(m^2)$ операцій множення.
3. Обчислення поліному в точці:
   $p(x) = d_0 + d_1(x - x_0) + d_2(x - x_0)(x - x_1) + ... + d_m(x - x_0)(x - x_1)...(x - x_{m-1}) =$
   $= (...(((d_m(x - x_{m-1}) + d_{m-1})(x - x_{m-2}) + d_{m-2})(x - x_{m-3}) + d_{m-3})... + d_1)(x - x_0) + d_0$
   $\Rightarrow$ m множень