

SPb HSE, 1 курс, осень 2022/23  
Практика по алгоритмам #9

Динамика-1

24 ноября

Собрано 2 декабря 2022 г. в 13:25

---

Содержание

1. Динамика-1	1
2. Разбор задач практики	3
3. Домашнее задание	7
3.1. Обязательная часть . . . . .	7
3.2. Дополнительная часть . . . . .	9

# Динамика-1

## 1. Рюкзак с ценами

Даны  $n$  предметов. У каждого есть цена  $v_i$  и вес  $w_i$ .

Найти max стоимость, которую можно набрать предметами суммарного веса  $\leq S$ .

(a) Время  $\mathcal{O}(nS)$ , память  $\mathcal{O}(S)$ .

(b) Два рюкзака размера  $S$ . Время  $\mathcal{O}(nS^2)$ , память  $\mathcal{O}(S^2)$ .

## 2. Ленивость

Есть  $n \leq 10^{18}$  котят. Нужно посчитать число групп, на которые поделятся котята, если они делятся на группы  $\lfloor \frac{n}{2} \rfloor$  и  $\lceil \frac{n}{2} \rceil$  при  $n > k$  и не делятся при  $n \leq k$ .

Доказать, что при решении ленивой динамикой будет  $\mathcal{O}(\log n)$  состояний.

## 3. Разбиение на палиндромы

Разбить строку на минимальное число палиндромов за  $\mathcal{O}(n^2)$  времени и  $\mathcal{O}(n)$  памяти.

## 4. Задача Иосифа Флавия

Древняя игра.  $n$  человек стоят по кругу. Считалочка. Каждый  $p$ -й по счету человек покидает круг и сбрасывается в яму с крокодилами. Даны  $n$  и  $p$ , кто останется в кругу последним?  $\mathcal{O}(n)$ .

Пример  $n = 6, p = 2$ . Люди уходят в порядке 2, 4, 6, 3, 1.

## 5. Редакционное расстояние

Даны две строки. За минимальное число операций «удаления одного символа», «вставки одного символа» и «замены символа на другой символ» получить из первой строки вторую.  $\mathcal{O}(nm)$  времени и памяти с восстановлением, где  $n$  и  $m$  – размеры строк.

## 6. Наибольшая общая возрастающая подпоследовательность

Даны  $a, b$ . Найти наибольшую общую возрастающую подпоследовательность  $a$  и  $b$ .

a)  $\mathcal{O}(n^4)$ .

b)  $\mathcal{O}(n^3)$ .

c) (\*)  $\mathcal{O}(n^2)$ .

## 7. Хиршберг

Придумайте решение за  $\mathcal{O}(n^2)$  времени и  $\mathcal{O}(n)$  памяти для задачи «найти **путь** по матрице вправо-вверх с максимальной суммой».

## 8. Строки Фибоначчи

a) Сколько существует строк длины  $n$  из 0, 1, не содержащих два нуля подряд?

b) Сколько существует строк длины  $n$  из 0, 1, 2, не содержащих два нуля подряд?

c) Посчитаем  $2^n$  по модулю  $m$ .  $n \leq 10^{18}$ ,  $m \leq 10^9$ .

d) Разгоним нашу динамику до  $\mathcal{O}(\log n)!$  (решим для  $n = 10^{18}$ )

## 9. Рекуррентное соотношение

Есть следующее рекуррентное соотношение:

$$\begin{aligned}a_n &= a_{n-1} + 2c_{n-1} + 1 \\b_n &= 5 - c_{n-1} \\c_n &= c_{n-2} - b_{n-1}\end{aligned}$$

Нам известны  $a_0, a_1, b_0, b_1, c_0, c_1$ . Найти  $a_n, b_n, c_n$  по модулю  $(10^9 + 7)$  за  $\mathcal{O}(\log n)$ .

## 10. Бешенный конь

Сколько способов пропрыгать конём из клетки  $(x_1, y_1)$  в клетку  $(x_2, y_2)$  за ровно  $k$  шагов. Решите для  $k \leq 10$ . А для  $k \leq 10^9$ ?

## 11. Ровный абзац

Дан текст (набор слов) с длинами  $l_1, l_2, \dots, l_n$ . Разбить текст на строки длины не более  $L$ . Менять порядок слов и переносить слова нельзя. Между каждой парой слов стоит хотя бы один пробел, остаток строки заполнен пробелами.

Минимизировать  $\sum \text{gap}_i^3$ , где  $\text{gap}_i$  – число пробелов в строке  $i$ .  $\mathcal{O}(nL)$ .

## 12. (\*) Задача о министерстве

Есть таблица  $n \times m$  клеток, в каждой клетке натуральное число. Найти путь min веса из  $(1, 1)$  в  $(n, m)$ , если можно ходить вверх, вправо и влево.  $\mathcal{O}(nm)$ , восстановить ответ.

a)  $\mathcal{O}(nm)$  памяти.

b) Работает ли Хиршберг? Вернее «за сколько работает»?

c) (\*)  $\mathcal{O}(nm^{1/2})$  памяти.

d) (\*)  $\mathcal{O}(nm^{1/3})$  памяти.

## 13. (\*) Точки на круге

Найти число подмножеств из  $n$  точек на круге, таких, что любые два соседних элемента множества на расстоянии 1, 3 или 5. Решить для  $n \leq 10^9$ .

## 14. (\*) Плохая подстрока

Сколько существует строк из 0, 1, 2, не содержащих, как подстроку  $s$ ,  $|s| \leq 50$ ?

## 15. (\*) Текст в окне

Дан текст из  $n$  слов. Выбрать подотрезок слов, который поместится на экран  $h \times w$  без переносов, и суммарная длина слов в котором максимальна.

## 16. (\*) Наибольшая общая возрастающая подпоследовательность

$\mathcal{O}(n^2)$ ,  $\mathcal{O}(n)$  памяти + восстановление ответа. Докрутить Хиршберга.

## 17. (\*) Быстрый калькулятор

Получить из числа 1 число  $N$  операциями  $+1$ ,  $*2$ ,  $*3$ . Минимизировать число операций. Решить для  $N \leq 10^{18}$ . Корректность и время работы решения требуется обосновать.

# Разбор задач практики

## 1. Рюкзак с ценами

```

1 for (int i = 0; i < n; ++i)
2     for (int j = S; j >= w[i]; --j)
3         f[j] = max(f[j], f[j - w[i]] + v[i]);
4 answer = f[S];

```

Если есть два рюкзака, то будет  $f[j_1, j_2]$ .

И три перехода – скипнуть, кинуть в первый, кинуть во второй:

$$f[j_1, j_2] = \max(f[j_1, j_2], f[j_1 - w_i, j_2] + v_i, f[j_1, j_2 - w_i] + v_i);$$

## 2. Ленивость

Решение = рекурсия с запоминанием! Состояний будет  $2 \log n$ . Доказательство:

На каждом уровне рекурсии работаем с числами из отрезка  $[L_i, R_i]$ .  $L_1 = R_1 = n$ .

Утверждение:  $R_i - L_i \leq 1$ . Переход:  $[2k, 2k + 1] \rightarrow [k, k + 1]$ ,  $[2k - 1, 2k] \rightarrow [k - 1, k]$ .

## 3. Разбиение на палиндромы

Решение за  $\mathcal{O}(n^2)$  времени.

$f[i]$  – кол-во палиндромов, на которые можно разбить префикс.

$$f[i] = \min_{j < i, [j, i] \text{ - палиндром}} (f[j] + 1).$$

Есть два способа быстро проверить, является ли строка палиндромом:

- Предподсчитать для всех пар  $[j, i]$  за  $\mathcal{O}(n^2)$ , но это требует квадрат памяти.
- Для каждого  $i$  найти  $r_0[i]$ ,  $r_1[i]$  – длины максимальных палиндромов чётной и нечётной длины с центром в  $i$ .

## 4. Задача Иосифа Флавия

Когда из круга уйдёт один человек, останется круг из  $n - 1$  человека. В этом круге последним уйдёт человек номер  $f[n - 1]$ . В исходном круге он имел номер  $f[n] = (f[n - 1] + p) \% n$ .

## 5. Редакционное расстояние

Разобрана в конспекте.

## 6. Наибольшая общая возрастающая подпоследовательность (НОВА)

- Простейшее решение:  $f[i, j]$  – max длина НОВА, заканчивающейся ровно в  $a[i]$  и  $b[j]$ .

Переход динамики вперёд: выбрать  $i_1 > i$  и  $j_1 > j$  такие, что  $a[i_1] = b[j_1]$ ,  $a[i_1] > a[i]$ .

С ходу получили  $\mathcal{O}(n^4)$ .

Можно оптимизировать до  $\mathcal{O}(n^3)$ : делать переход в два этапа  $f[i, j] \rightarrow g[i_1, j] \rightarrow f[i_1, j_1]$ .

- Идея на  $\mathcal{O}(n^2)$  – правильно выбрать состояние.

$ans[j, i]$  – длина НОВА, кончающейся **ровно** в  $a[i]$ , и **строго раньше**  $b[j]$ .

Два перехода в динамике вперёд: пытаемся взять или не взять  $b[j]$ .

Если не берём, попадаем в  $ans[j + 1, i]$ , если берём, попадаем в  $ans[j + 1, next(i, b[j])]$ ,

где  $next(i, b[j])$  – ближайший за  $a[i]$  элемент, равный  $b[j]$ .

Функция  $next$  для каждого  $b[j]$  при увеличении  $i$  насчитывается вторым указателем.

- Можно последнюю идею написать ещё проще динамикой назад.

Будем пересчитывать по слоям:  $ans[j] \rightarrow ans[j + 1]$ .

Изначально  $ans[j+1] = ans[j]$ , добавляем  $b[j]$ . Перебираем  $i = 0..|a|-1$ , когда приходим в  $a[i] = b[j]$ , чтобы получить  $ans[j+1, i]$  нужно помнить  $F = \max_{k: a[k] < b[j], k < i} ans[j, k]$ .

```

1 F = 0, ans[j+1] = ans[j]
2 for i=0..|a|-1:
3     if a[i] < b[j]: F = max(f, ans[j, i])
4     if a[i] = b[j]: ans[j+1, i] = max(ans[j+1, i], F+1)

```

Ну и напоследок заметим: достаточно хранить только одну строку.

## 7. Хиршберг

Обычный Хиршберг (см.конспект) — запускаем с двух сторон исходную динамику, получаем столбцы  $f[j], g[j]$ , выбираем  $j: f[j] + g[j] = \max$ , запускаемся рекурсивно от двух подматриц.

## 8. Строки Фибоначчи

- Сколько существует строк длины  $n$  из  $0, 1$ , не содержащих два нуля подряд?  
В конце строки либо 1, либо 10  $\Rightarrow f_n = f_{n-1} + f_{n-2}$ .
- Сколько существует строк длины  $n$  из  $0, 1, 2$ , не содержащих две равные цифры подряд?  
В конце строки либо 1, либо 2, либо 10, либо 20.  $f_n = 2(f_{n-1} + f_{n-2})$ .
- Посчитаем  $2^n$  по модулю  $m$  за  $\mathcal{O}(\log n)$  умножений.

```

1 int pow(int a, int n):
2     if (n == 0) return 1;
3     int res = pow(a * a % m, n / 2);
4     return n % 2 ? a * res % m : res;

```

- Таким образом возводить в степень можно всё, что угодно. Например  $f_n = 2(f_{n-1} + f_{n-2}) \Rightarrow$

$$\text{можно посчитать за } \mathcal{O}(\log n): \begin{bmatrix} f_{n+1} \\ f_n \end{bmatrix} = \begin{bmatrix} 2 & 2 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} f_n \\ f_{n-1} \end{bmatrix} = A \cdot \begin{bmatrix} f_n \\ f_{n-1} \end{bmatrix} = A^n \cdot \begin{bmatrix} f_1 \\ f_0 \end{bmatrix}.$$

Умножение матриц ассоциативно  $\Rightarrow$

сперва считаем  $B = A^n$  алгоритмом выше, затем  $f_n = B_{00}f_1 + B_{01}f_0$ .

## 9. Рекуррентное соотношение

Запишем матрицу, выражающую  $\langle a_n, b_n, c_n, c_{n-1} \rangle$  через предыдущие.

$$\begin{bmatrix} a_n \\ b_n \\ c_n \\ c_{n-1} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 & 0 & 1 \\ 0 & 0 & -1 & 0 & 5 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} a_{n-1} \\ b_{n-1} \\ c_{n-1} \\ c_{n-2} \\ 1 \end{bmatrix}$$

## 10. Бешенный конь

$f[k, x, y]$  — число способов дойти из  $(x_0, y_0)$  до  $(x, y)$  за  $k$  шагов.

$$f[k, x, y] = \sum_{dx, dy} f[k-1, x-dx, y-dy] \text{ решит } k \leq 10.$$

Получили решение за  $k \cdot 64 \cdot 8$  (из каждой клетки 8 прыжков коня).

Для больших  $k$  можно быстрее. Переходить не  $k \rightarrow k+1$ , а  $k \rightarrow 2k$ :

$$f[2k, x_1, y_1, x_3, y_3] = \sum_{x_2, y_2} f[k, x_1, y_1, x_2, y_2] \cdot f[k, x_2, y_2, x_3, y_3].$$

Здесь  $f[k, a, b, c, d]$  — число путей длины  $k$  из  $(a, b)$  в  $(c, d)$ . Если обозначить за  $f_k$  двумерный массив  $64 \times 64$ , выше написано  $f_{2k} = f_k \cdot f_k$ , умножение матриц. Воспользоваться оптимизацией  $k \rightarrow 2k$  мы можем и не зная ничего ни про матрицы, ни про их умножение.

## 11. Ровный абзац

Примерно так выравнивают тексты всякие `word` и `tex`.

$f[i]$  – стоимость разбить первые  $i$  слов.

Переход: перебираем  $j \uparrow$ , все слова из  $[i, j)$  пишем в одну строку...

## 12. (\*) Задача о министерстве

$f[i, j]$  – минимальный путь в  $(i, j)$ .  $f[i] \rightarrow f[i + 1]$ :

a) Сначала  $f[i+1, j] = f[i, j] + \text{cost}[i, j]$ .

b) Проходим слева направо и  $\text{relax}(f[i+1, j], f[i+1, j-1] + \text{cost}[i, j])$ .

c) Проходим справа налево и  $\text{relax}(f[i+1, j], f[i+1, j+1] + \text{cost}[i, j])$ .

**Оптимизируем память:** Хиршберг не работает! ему нужна монотонность и по  $i$ , и по  $j$ .

Но мы можем сохранить каждую строку матрицы с шагом  $m^{1/2}$ , а затем во время восстановления ответа на каждом из  $m^{1/2}$  кусков за  $\mathcal{O}(nm^{1/2})$  насчитать динамику ещё раз.

А ещё можно сохранять куски с шагом  $m^{2/3}$  (чтобы было  $nm^{1/3}$  памяти).

Тогда восстановление ответа –  $m^{1/3}$  вызвать предыдущую идею для матрицы размера  $nm^{2/3}$ .

## 13. (\*) Точки на круге

Это упражнение на тему «узрите матрицу, возведите её в степень».

Как догадаться, что где-то здесь нужно возведение в степень?

$n = 10^9$ , поэтому решение – или формула, или динамика с возведением матрицы в степень.

Решение:  $f[i] = f[i - 1] + f[i - 3] + f[i - 5]$ .

Линейное рекуррентное соотношение, размер матрицы  $5 \times 5$ , обозначим матрицу  $A$ . У нас круг, надо это учесть. Заметим, что расстояние между первой и последней точкой –  $n - 1$ ,  $n - 3$  или  $n - 5 \Rightarrow \text{res} = A^{n-1}[0, 0] + A^{n-3}[0, 0] + A^{n-5}[0, 0]$ .

**P.S.** В общем случае при решении задачи на круге нужно было бы перебрать все корректные состояния  $v$  первых пяти точек и посчитать  $\sum_v B^n[v, v]$ , где  $B$  – матрица переходов между «состояниями 5 подряд идущих точек».

## 14. (\*) Плохая подстрока

$f[i, j]$  – число строк длины  $i$ , таких, что наибольший суффикс совпадающий с префиксом  $s$  имеет длину  $j$ . Если знаем Ахо-Корасик, можно ещё и его сверху, чтоб быстрее было.

## 15. (\*) Текст в окне

a) Двумя указателями посчитали  $\text{next}[i]$  – конец строки, начатой в  $i$ -м слове.

b) На полученном дереве  $i \rightarrow \text{next}[i]$  или двоичные подъёмы, или решаем LA.

## 16. (\*) Наибольшая общая возрастающая подпоследовательность

```

1 for j=0..|b|-1
2   F = 0;
3   for i=0..|a|-1:
4     if a[i] < b[j]: F = max(f, ans[i])
5     if a[i] = b[j]: ans[i] = max(ans[i], F+1)

```

Насчитаем динамику с двух сторон встретимся на строке  $|a|/2$ .

Выберем  $j$ :  $\text{ansLeft}[j] + \text{ansRight}[j] = \max$ . Точно возьмём элемент  $a[j]$ . Выделим подзадачи  $b[0, \frac{n}{2}) \times a[0, j)$  и  $b[\frac{n}{2}, |b|) \times a[j, |a|)$ . Обе задачи отфильтруем по  $a[j]$ . Запутимся рекурсивно.

17. (\*) **Быстрый калькулятор**

А давайте идти не от 1 к  $N$ , а от  $N$  к 1. И ленивости добавим.

Если бы не было +1, было бы совсем мало достижимых, динамика уже работала бы на ура.

Заменяем динамику на **bfs**.

Утверждение #1: ответ не больше  $k = 2 \log n$  (используем только +1 и \*2).

Утверждение #2: мы посетим не более  $\binom{k}{3} = \mathcal{O}(\log^3 n)$  состояний.  $k = a + b + c$ ,  $N \rightarrow \frac{N}{2^a 3^b} - c$ .

# Домашнее задание

## 3.1. Обязательная часть

### 1. (4) Сводный брат Фибоначчи

Мы стоим в точке 0, за ход можем сместиться вправо на любое из чисел  $1, 2, \dots, k$ .  
Сколько способов прийти в точку  $n$ ?

- a) (1)  $\mathcal{O}(nk)$
- b) (1)  $\mathcal{O}(n)$
- c) (2)  $\mathcal{O}(\text{poly}(k) \log n)$

### 2. (4) Рюкзак со стоимостями

Вес. Стоимости. Время  $\mathcal{O}(nS)$ , память  $\mathcal{O}(S)$ .

Набрать вес ровно  $S$ , максимизировать стоимость выбранных предметов.

- a) (1) Каждый предмет можно брать один раз.
- b) (1) Каждый предмет можно брать сколько угодно раз.
- c) (1) Каждый предмет можно брать сколько угодно раз.  
Восстановление ответа. Простым эффективным способом.
- d) (1) Каждый предмет можно брать один раз + восстановление ответа.

### 3. (2) Равномерное разбиение массива

Разбить массив длины  $n$  на  $k$  отрезков так, чтобы сумма квадратов сумм отрезков была минимальна. Пример:  $n = 4, k = 3, a = \{7, 2, 1, 4\} \rightarrow \{7\} + \{2, 1\} + \{4\} \rightarrow 7^2 + (2+1)^2 + 4^2 = 74$ .

### 4. (2) Наибольшая пилообразная подпоследовательность

Последовательность называется пилообразной, если никакие её три подряд идущих элемента не образуют ни возрастающую, ни убывающую последовательность. Найдите наибольшую пилообразную подпоследовательность данного массива за  $\mathcal{O}(n^2)$ .

(+1) балл за  $\mathcal{O}(n^{2-\varepsilon})$ .

### 5. (3) LCP

Дана строка  $s$ . Для каждой пары  $(i, j)$  найти длину общего префикса  $i$ -го и  $j$ -го суффиксов  $s$ .  
Пример: для  $s = \text{ababaababaabb}$  имеем  $\text{LCP}(2, 7) = 5$ .

$s = \text{ababaababaabb}$

$s = \text{ababaababaabb}$

### 6. (4) LZSS

Дана строка из латинских букв длины  $n$ , нужно ее запаковать в максимально короткую, используя правило  $(n, i)$  — повторить  $n$  символов начиная с  $i$ -й позиции.

Например,  $s = \text{xyabababababz} \rightarrow \text{xyab}(8, 2)z$ .

Еще пример:  $s = \text{xyaaaaabaaaabaab} \rightarrow \text{xya}(3, 2)b(10, 2)$ . Более оптимально  $\text{xyaaaaab}(10, 2)$ .

*Переверните страничку, там дальше разбалловка.*

Как кодировать  $(10, 2)$ ? можно так и писать строкой длины 6, длина строки зависит от  $n$  и  $i$ , не константа. А можно сказать, что  $n$  и  $i$  32-битные  $\Rightarrow \forall n, i$  ровно 8 байт.

(2) Решить за  $\mathcal{O}(n^3)$ . Длина  $(n, i)$  – константа.

(4) Решить за  $\mathcal{O}(n^2)$ . Длина  $(n, i)$  – константа (не бойтесь быть жадными!)

(+1) Решить за  $\mathcal{O}(n^2)$ . Длина  $(n, i)$  – **не** константа.

P.S. В этой задаче описан крутой реально работающий архиватор. Аккуратная реализация даёт уровень сжатия `rar/zip`. В третьем семестре мы научимся реализовывать его за  $\mathcal{O}(n)$ .

## 3.2. Дополнительная часть

### 1. (3) Рюкзак с мультипредметами

Даны  $n$  предметов. У каждого есть цена  $v_i$  и вес  $w_i$ . Каждый предмет можно взять от 0 до  $k_i$  раз. Найти максимальную суммарную цену, которую можно набрать таким образом, чтобы суммарный вес не превышал  $W$ .  $\mathcal{O}(nW)$ .

### 2. (4) Перестановки с локальными минимумами

Посчитайте количество перестановок из  $n$  элементов с ровно  $k$  локальными минимумами.

a) (2)  $\mathcal{O}(n^3)$

b) (2)  $\mathcal{O}(n^2)$

### 3. (4) Выражение

Даны натуральные числа и знаки  $+$ ,  $*$  между ними. Есть ли порядок выполнения операций, чтобы получить требуемое число  $X$ ? Открытые ограничения.

Интересны лишь решения, для  $X \sim 10^9$  и больше.

### 4. (3) Быстрый НОП

Найдите наибольшую общую подпоследовательность за  $o(n^2)$ . Элементы из  $[1..n]$ .