# Алгоритмы. Домашнее задание №6

Горбунов Егор Алексеевич

23 октября 2015 г.

**Задача №1** (Число строк свободных от $s$)

**Задача №2** (Взвешенный бинарный поиск)

Нужно определить стоимость поиска элемента в массиве $A[1..n]$ в худшем случае, даны стоимости обращения к элементу $i$: $c_i$. Введём:

$$D[i][j] \text{ — стоимости поиска в худшем случае в подмассиве A[i..j]}$$

Ясно, что $D[i][i] = c_i$. При поиске в $A[i..j]$ мы можем выбрать разделяющим любой элемент из $i..j$, т.е. подсчёт $D[i][j]$ таков:

$$D[i][j] = \max_{i \le k \le j} \left( \max \left( D[i][k-1], D[k][j] \right) + c_k \right)$$

Решение задачи после подсчёта находится в $D[1][n]$, для того, чтобы его найти нужно заполнить таблицу $D$ и для каждой её клетки затратить $\mathcal{O}(n)$ операций. Отсюда получаем сложность $\mathcal{O}(n^3)$. Заполнение $D$ идёт по диагоналям, начиная с главной. ∎

**Задача №3** (Число способов расставить на доске...)

**Задача №4** (Расстояние до палиндрома)

Дана строчка $s[1..n]$, нужно превратить строчку $s$ в палиндром за минимальную суммарную стоимость операций, если возможны операции: удаление символа стоимостью в $A$ и замена символа стоимостью в $B$.

Введём следующую величину:

$D[i][j]$ — наименьшая суммарная стоимость операций по приведению подстроки s[i..j] к палиндрому

$D[i][i] = 0$, т.к. 1 символ — палиндром. $D[i][j] = 0, j < i$.

1. Если в оптимальном приведении $s[1..n]$ к палиндрому первый и последний символы $s$ остались нетронутыми, т.е. $s[1] = s[n]$, то стоимость оптимального приведения $s[1..n]$ к палиндрому равно стоимости оптимального приведения к палиндрому $s[2..n-1]$.
2. Если в оптимальном приведении $s[1..n]$ к палиндрому $s[1]$ и $s[n]$ были удалены, то оптимальная стоимость приведения к палиндрому $s[1..n]$ равна $D[2][n-1]+2A$ (дважды удалили)
3. ... $s[1]$ был заменён на $s[n]$ (или наоборот), то $D[1][n] = D[2][n-1]+B$ (ясно, что $s[1] \neq s[n]$ в данной ситуации)
4. ... $s[1]$ был удалён (а $s[n]$ нет), то $D[1][n] = D[2][n] + A$ и аналогично, если $s[n]$ был удалён, а $s[1]$ нет.

Других случаев быть не может и таким образом получили следующее решение методом динамического программирования:

$$D[i][j] = \min \begin{cases} D[i+1][j-1] & , s[i] = s[j] \\ D[i+1][j-1] + B & , s[i] \neq s[j] \\ D[i+1][j-1] + 2A & \\ D[i+1][j] + A & \\ D[i][j-1] + A & \end{cases}$$

Чтобы вычислить ячейку $D[i][j]$ нужно знать значения в ячейках левее и ниже. Таким образом можно просчитывать $D$ по диагоналям. Ответ будет записан в правой верхней ячейке $D[1][n]$ и найдём его за время $\mathcal{O}(n^2)$. ■

## Задача №5 (-)

## Задача №6 (Про палиндромы...)

(a) Найти самую длинную подпоследовательность-палиндром за $\mathcal{O}(n^2)$

Мы умеем за $\mathcal{O}(n)$ превращать удалениями символов и заменой символов строку $s[1..n]$ в палиндром (это задача №4). Заметим тогда, что если положить стоимость удаления символа $A = 1$, а $B = (n+1)$ — стоимость замены,то алгоритм построение преобразования строки $s$ в палиндром всегда будет пользоваться лишь операцией удаления, а т.к. он находит такое редактирование, что суммарная стоимость операций минимальна, т.е. число удалений минимально, то мы получим в результате наибольшую подпоследовательность-палиндром. ■

(b) Посчитать число подстрок палиндромов за $\mathcal{O}(n^2)$

Будем идти по строке $s[1..n]$ слева направо. На каждой итерации рассматриваем $i$-ый элемент строки и делаем следующее:

1) пытаемся построить палиндром нечётной длины с центром в $s[i]$:

$l \leftarrow i - 1$

$r \leftarrow i + 1$

**while** $l \geq 1$ AND $r \leq n$ AND $s[l] = s[r]$ **do**

$l--;\ r++;$

$polindromeCnt \mathrel{+}= 1$

2) пытаемся построить палиндром чётной длины с центром «между» $s[i-1]$ и $s[i]$ $(i > 1)$, псевдокод аналогичен тому, что приведён выше.

Ясно, что т.к. мы перебираем центры палиндромов и все они различны, то ни один палиндром не будет посчитан дважды, а так же все палиндромы будут посчитаны, сложность $\mathcal{O}(n^2)$

(c) Разбить строку на минимальное число палиндромов за $\mathcal{O}(n^2)$

За квадратичное время от длины строки, по пункту $(b)$ этой задачи мы можем найти все палиндромы. Их, очевидно, уж точно не более $n^2$. Тогда мы можем за квадратичное время построить массив $P[1..n]$, что в $P[i]$ хранятся все палиндромы, которые начинаются на позиции $i$ строки $s$. Будем считать, что мы за $\mathcal{O}(1)$ можем находить длину палиндрома (строки). Тогда ясно, что в любом разбиении строки $s$ на палиндромы (в том числе и минимальном), какой-то палиндром стоит в начале с позиции 1 по какую-то позицию $k$. Будем находить тогда величину $D[i]$ — минимальное число палиндромов в разбиении $s[i..n]$. $D[n] = 1$.

$$D[i] = \min_{s[i..k] \in P[i]} (1 + D[k+1])$$

Ответ будет находиться в ячейке $D[1]$ и найден он будет за $\mathcal{O}(n^2)$ (включая нахождение всех палиндромов вначале). ■