ДП.

вперёд | назад

bitset

N: |10101... | | | $O\left(\frac{N}{w}\right)$

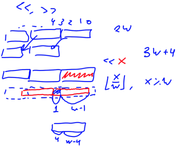
N = 32: W

int

uint32_t a[N/w]

б

bitset: &, |, ^


<<, >>
4 3 2 1 0
2w
1
3w+4
<< x
⌊x/w⌋ , x ≥ w
1
w-1
4
w-4

for i = 1..n

Bts |= (Bts << x_i)

$O(\frac{N}{w}) = O(N)$

w = 64

$w \geq \log N$ 　 log(InputSize)

N

| log N |
addr

$x_1 \ldots x_N$

$[0, 2^w)$

НОП. $O(n^2)$

$O(n^2/w)$ space time

$S_i = t_j$

$O(n)$ space

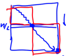
$-dp[i][j]$

$dp[i][j] \le dp[i][j+1] \le$
$\le dp[i][j]+1$

$dp[i][j+1] - dp[i][j] \in \{0, 1\}.$

n^2 bit
n^2/w слов

Хиршберг



<dp, Mark>

f[i][j].Mark = j

f[n][m].Mark

O(m)

dp_i

dp_{i+1}

for: vector<...> dp(M), dp_new(M)

dp_new считается через dp

swap(dp, dp_new)

НОП, про путь по матрице, стоимостной рюкзак

Способ 2 (Хиршберг)

Доп. (LCS)

$lcs(s_1 \ldots s_n, t_1 \ldots t_m) = \exists j:$ a_j

$\quad n/2 \quad n/2 \quad j \quad m-j$

$= lcs(s_1 \ldots s_{n/2}, t_1 \ldots t_j)$

$+ lcs(s_{n/2+1} \ldots s_n, t_{j+1} \ldots t_m)$

b_j $\quad lcs(s_n \ldots s_{n/2+1}, t_m \ldots t_{j+1})$

$j = ?$

$(n, m) \to (\frac{n}{2}, j) + (\frac{n}{2}, m-j) \qquad O(nm)$

$a_j = lcs(s_1 \ldots s_{n/2}, t_1 \ldots t_j)$

$b_j = lcs(s_{n/2+1} \ldots s_n, t_{j+1} \ldots t_m)$

$lcs(x, y) = lcs(rev(x), rev(y))$

HBN (LIS)

$O(n^2)$: $dp_i = 1 + \max_{\substack{j<i \\ a_j<a_i}} dp_j \quad (\max(\emptyset)=0)$

$par_i = j$

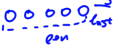
$O(n \log n)$

$dp[i][j] = Func(i, j)$

Процесс

< i, len, last > → a_i

f[i][len][i] = 0/1



len[i, last] = max len:

len[i, j]

<i, len, last>

$\langle i, len, last \rangle$ $\quad a_i$ $\quad$ | 1 -- i | |

$last[i, len]$ = min значение $last$, $O(n^2)$

это $\langle i, len, last \rangle$

$last_i[len]$

$$\begin{matrix} 1 & 2 & 3 \\ a_1 < a_2 < a_3 \dots \end{matrix}$$

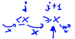
$x < y < a_3$

x, y

10, 20, 30

10, 20

$a_2 < y$ | .., 53, 30

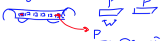


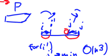
last = [-∞, +∞, +∞, ----]

for x in arr:

* lower_bound(last.beg(), last.end()) = x;

Погрузка кораблей





dp[i][j] = осталась отрезок [i, j], сколько кораблей понадобится чтобы его отправить.

dp[1][n] = Ans

dp[пустого отрезка] = 0

$dp[i][j] = 1 + \min_{i',j'} dp[i'+1][j'-1]$

$O(n^4)$

<i, j, full, w>

1) dp[i][j][full][w] = 0/1

2) W[i, j, full] = min W:

<i, j, full, w>

i+1, full+1, j-1

$O(n^3)$

<i, j, full, w> <3, 0> full → min, w → min

w [i, j, full].

<full = 2, w = 10>

<full = 3, w = 1>

dp[i, j] = min <full, w> :

<i, j, full, w>

full

<i, j, x, y>

x → min

y → min

dp[i, j] = min <x, y>

ДР и Матрицы

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} f_n \\ f_{n-1} \end{bmatrix} = \begin{bmatrix} f_{n+1} \\ f_n \end{bmatrix}$$

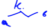
$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = A$

$$A^n \begin{bmatrix} f_1 \\ f_0 \end{bmatrix} = \begin{bmatrix} f_{n+1} \\ f_n \end{bmatrix}$$

$k^3 \log n$

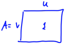
$k^2 \log n$, $k \log k \log n$

Графы и матрицы.



$$dp_{v,len} = \begin{cases} 1, & len = 0 \\ \sum_{(u,v)\in E} dp_{u,len-1} & len > 0 \end{cases}$$

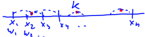
$$Ans = \sum_v dp_{v,k}$$

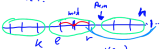


$$dp[len=0] = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \qquad dp[len=k] = A^k \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

$$dp[len=1] = A \cdot dp[len=0]$$

Почтовые Отделения. (сервера)





$$cost(l,r) =$$

$$= \sum_{i=l}^{r} |x_i - x_{med}| =$$

$$(l+r)/2$$

$$= \sum_{i=l}^{med} x_{med} - x_i + \sum_{i=med+1}^{r} x_i - x_{med}$$

$$= x_{med} \cdot C + \sum_{i=l}^{med} x_i + \sum_{i=med+1}^{r} x_i$$

Оптимизация Кнута

$dp_{k,n}$ = разбить $x_1 \dots x_n$ на k кусочков.

$$dp_{0,0} = 0$$

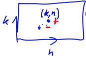
$$dp_{k,n} = \min_{l=1..n} dp_{k-1,l-1} + cost(l,n)$$

$$P_{k,n} = l$$

$$P_{k-1,n} \leq P_{k,n} \leq P_{k,n+1}$$

```
for k = 1..K
  for n = N..1
    for l = 1..n        B = P_{k-1,n} .. P_{k,n+1}
      dp_{k,n} ← ...
```

$$T = O(nk) + \underbrace{\sum_{k,n} p_{k,n+1} - p_{k-1,n}}_{O(n^2)}$$



$O(n+k) = O(n)$

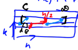
$$T = O(n^2)$$

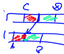
Опт. кнута.

Розд и властвуй

$O(nk \log n)$ $dp_{k,n} \leftarrow dp_{k-1,\, l-1} + cost(l,n)$

• $p_{k,n} \le p_{k,n+1}$





$O((\log n)\cdot n)$

```
go(a, b, c, d):        ← if c>d: return
  m = (c+d)/2          // dp_{k+1,m}
  for i=a..b           ← пусть i* - опт. инд.
    if i<m:
      dp_{k+1,m} ← dp_{k,i} + cost(i+1,m)
  go(a, i*, c, m-1); go(i*, b, m+1, d)
```