

Предшественик от определено ниво (част 3)

16.10.2020 г.

(Level Ancestor Query)

Колко са наредените коренови дървета от n -върха.

Задача: Нека $n \in \mathbb{N}$. Колко са думите $w \in \{0,1\}^{2n}$, за които

1. $|w|_0 = |w|_1$
2. за всеки префикс u : $|u|_0 \geq |u|_1$?

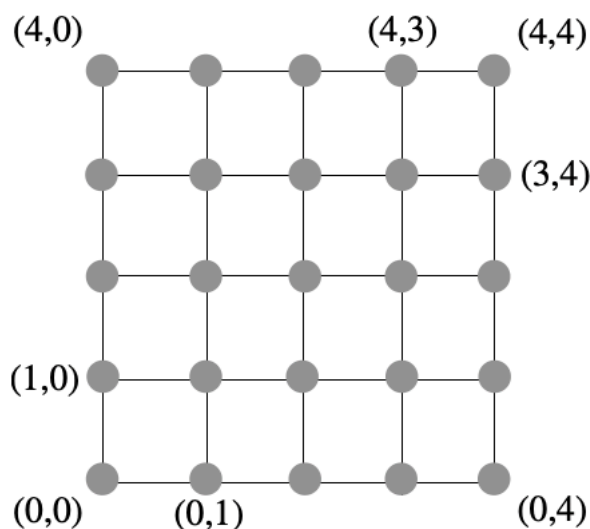
Задачата има следната еквивалентна формулировка:

(a) По колко начина може да стигнем от точката $(0,0)$ до точката (n,n) в стандартна квадратна решетка, ако правим единични стъпки само надясно и нагоре? А ако трябва да не преминаваме над диагонала, свързващ тези две точки?

(b) По колко коректни начина може да съставим дума от n откриващи и n закриващи скоби (пример: " $(())$ " е коректна, а " $()()$ " - не)?

Решение:

(a)



Нека разгледаме квадратната решетка за $n = 4$. Нейната размерност е 4×4 .

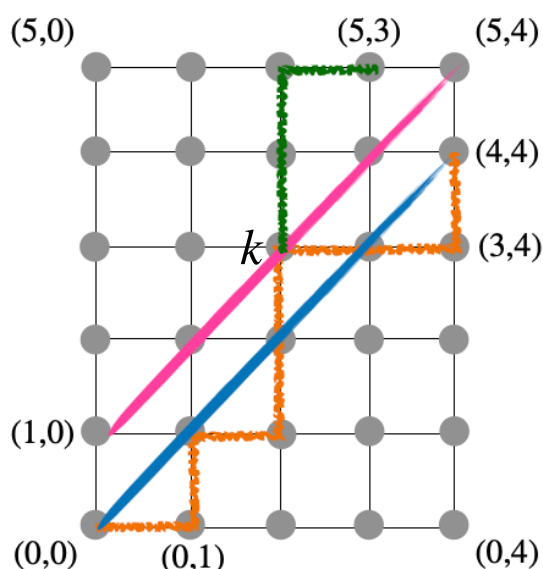
Очевидно за всяко n - пътя от началната до крайната точка ще е с дължина $2n$. Нека означим движение надясно с r , а движение нагоре с u .

Съществено е да направим следните наблюдения:

- Всеки един път от началната до крайната точка ще е от вида $\underbrace{rurruu \dots urru}_{2n}$
- Всички желани пътища ще имат еднаква дължина
- Ако R е множеството на поетите посоки надясно, а U е множеството на поетите посоки нагоре, то $|R| = |U| = n$.

Задачата се свежда до това да изберем n елементни подмножества на $2n$ елементно

множество. Това може да стане по $C_{2n}^n = \binom{2n}{n} = \frac{(2n)!}{(n)!(n)!}$ начина.



Нека сега преброим пътищата, които не минават над диагонала свързващ началната и крайната точка, спазвайки същите правила.

Броя на тези пътища ще намерим като от броя на всички пътища извадим броя на „лошите“ пътища, т.е. тези които си позволят да преминат през синия диагонал.

Нека разгледаме един такъв път, който е „лош“. Например оранжевият такъв от картинката по-горе. Освен това нека разширим решетката с още един ред нагоре. По този начин ще получим решетка от $n + 1$ реда и n колони. Сега, след като лошият път е пресякъл синия диагонал, то той или ще пресича и новия розов диагонал или най-малко ще има точка лежаща на него. От тази точка която лежи на розовия диагонал, до края на пътя ще направим симетричния му спрямо розовия диагонал и по този начин новия път (със зелената проекция и старото начало на оригиналния път) ще се намират изцяло в решетката с размерност $n + 1 \times n - 1$. Това може да го направим за всеки път, който си позволи да премине синия диагонал. Така между множеството на всички пътища в решетка с размерност $n + 1 \times n - 1$ и лошите (всички проектирани симетрично спрямо розовия диагонал) построихме биекция.

По този начин на всеки „лош“ път съпоставихме път от $(0, 0)$ до $(n - 1, n + 1)$. Обратно, ако имаме път от $(0, 0)$ до $(n - 1, n + 1)$, то той минава над главния диагонал и заменяйки r с u и u с r (т.е. \rightarrow с \uparrow и \uparrow с \rightarrow) след първия момент k , в който $x_k < y_k$ (където с x_k сме отбелязали броя на r (а с y_k броя на u) в катия момент/преход от пътя) получаваме „лош“ път от $(0, 0)$ до (n, n) , който минава над главния диагонал в момента k .

Но броя на всички пътища в решетка с размерност $n + 1 \times n - 1$ е

$$C_{2n}^{n-1} = C_{2n}^{n+1} = \binom{2n}{n-1}, \text{ например избираме } n-1 \text{ подмножества от } 2n \text{ елементно}$$

множество (или аналогично $n + 1$ подмножества от $2n$ елементно подмножество (биномните коефициенти от един ред са симетрични - *триъгълник на Паскал*))

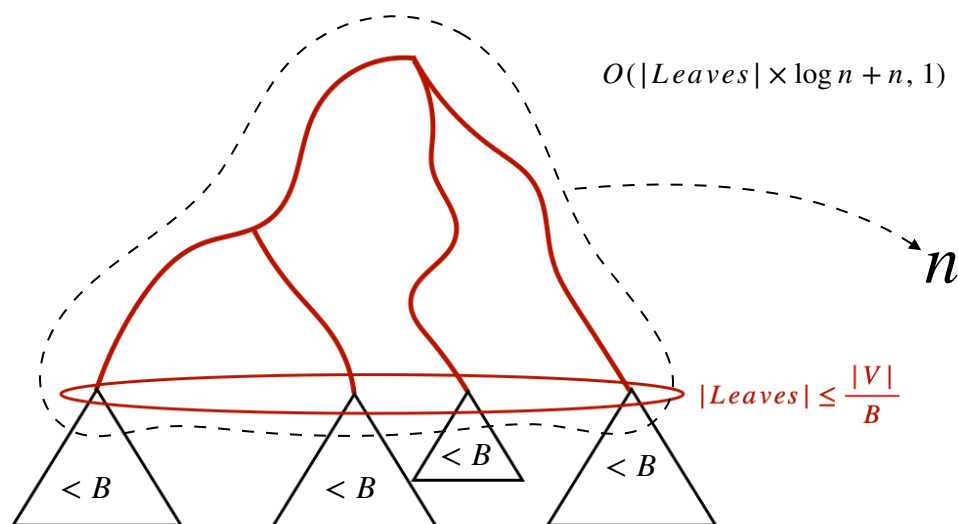
Следователно броя на търсените пътища е равен на

$$\binom{2n}{n} - |BAD| = \binom{2n}{n} - \binom{2n}{n-1} = \frac{(2n)!}{n!n!} - \frac{(2n)!}{(n-1)!(n+1)!} = \binom{2n}{n} \left(1 - \frac{n}{n+1}\right) = \frac{1}{n+1} \binom{2n}{n}$$

което е n -тото число на Каталан (https://en.wikipedia.org/wiki/Catalan_number).

(b) Във втората част на (a) подточка, може да направим следното наблюдение. Ако път който не пресича диагонала се характеризира с дума w , то в записа на тази дума няма как да се появят k броя букви u , ако преди това не е имало ПОНЕ толкова на брой букви r . Но това последно условие е еквивалентно на коректен запис от скоби, ако положим отваряща скоба да е r , а затваряща u . Тоест броя на коректните записи от отварящи и затварящи скоби е $\binom{2n}{n} - \binom{2n}{n-1}$.

Да се върнем към първоначалната задача. Стигнахме до следното дърво:



Идея: Отделяме малки дървета с размер по-малък от B .

Ако N е броя на върховете в цялото дърво, то $L \leq \frac{N}{B}$ (L е броя на листата в цялото дърво и $n \leq N$).

За червеното макро-дърво, решението ще бъде $O(n + L \times \log n, 1)$, като

$n + L \times \log n \leq N + \frac{N}{B} \log N$. Искаме да подберем B по такъв начин, че да е изпълнено:

$N + \frac{N}{B} \log N \leq c \cdot N$, за някаква константа c . Достатъчно е B да бъде от вида $B \geq \frac{\log N}{c_0}$,

за някоя константа c_0 , защото тогава $\frac{N}{B} \log N \leq \frac{N}{(\log N)/c_0} \times \log N = c_0 \times N$.

($\Rightarrow c = c_0 + 1$)

За микро-дърветата: За всяко микро-дърво пресмятаме думата

$w(T_v) \in \{0, 1\}^*$, $|w(T_v)| \leq 2(B-1)$.

$w(T_v) = \epsilon$ или $w(T_v) \in \{0, 1\}^*$ (регулярен израз) $\Rightarrow w(T_v)$ също съпоставя различни думи на неизоморфните наредени дървета.

$1 \circ w(T_v)$ число в двоична бройна система с $\leq 2B-1$ цифри $\Rightarrow 1 \circ (T_v) < 2^{2B}$ (ще използваме тези числа за индекс)

- I. За всяко число $\overline{1 \circ w(T_v)}$, T_v - микродърво, намираме едно конкретно дърво T_{v_0} такова, че $1 \circ (T_{v_0}) = 1 \circ (T_v)$ и пресмятаме решението $(n^2, 1)$ за T_{v_0} .
 $|T_{v_0}| < B$, т.е. това ще отнеме $(B^2, 1)$. Общия брой действия е $\leq \underbrace{2^{2B}}. B^2$, това е

защото имаме най-много толкова различни думи.

- II. За всяко микро-дърво T_v пресмятаме биекция между T_v и T_{v_0} със свойството, че $1 \circ w(T_{v_0}) = 1 \circ w(T_v)$ и T_{v_0} е било индексирано в стъпка I.

$$\begin{matrix} f_v \\ f_v^{-1} \end{matrix}$$

- III. За всеки микро-връх u съхраняваме следните неща:
1. Микро-корена $root(u)$ на микродървото, в което се намира u .
 2. Индекс на u в $f_{root(u)}$, където $f_{root(u)}$ е представено като масив.

Заявки: (v, d) , v е връх в оригиналното дърво

- I. сл. v е макро-връх, тогава се обръщаме към макро-дървото със заявка $LQ_{MACRO}(v, d)$, а времето е $O(1)$.
- II. сл. v е микро-връх. Тогава:
1. Намираме корена на неговото микро-дърво v'
 2. Проверяваме дали отговора е в микро-дървото: ако $d(v) - d(v') \geq d \Rightarrow$ отговора е в микро-дървото. Тогава намираме такова $v'_0 : T_{v_0}$ е индексирано и $w(T_v) = w(T_{v'_0})$
 3. Правим заявка в индекса за $v'_0 : u'_0 = (f'_v(v), d)$ **return** $f^{-1}(u'_0)$
 4. Ако $d > d(v) - d(v')$, тогава правим заявка към макро-дървото за следния връх и актуализирано разстояние: $LQ_{MACRO}(p(v'), d - 1 - d(v) + d(v'))$.

$\underbrace{\hspace{10em}}_{\text{разстояние от } v \text{ до } p(v') \text{ със знак минус}}$

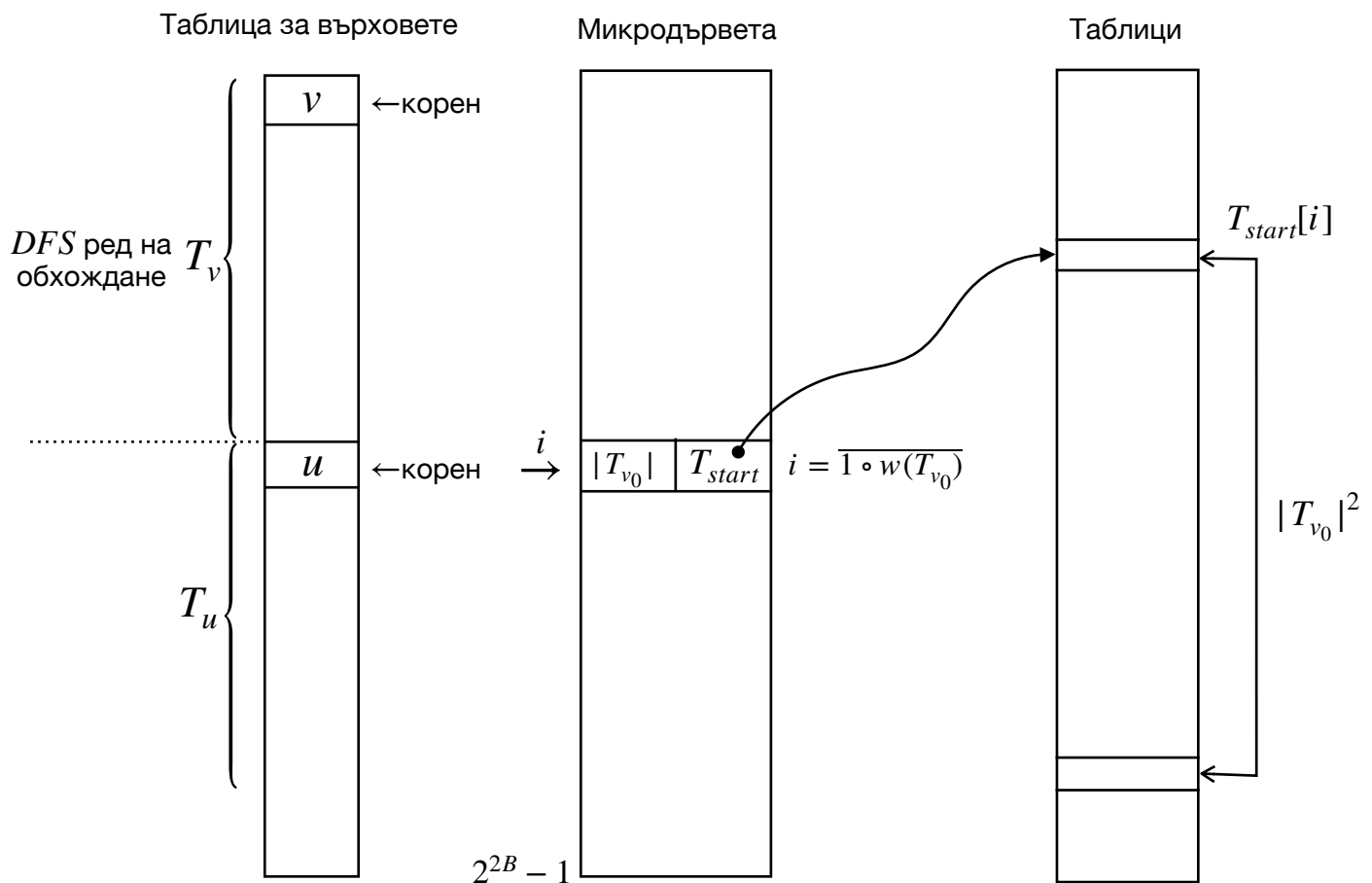
Така общото време за индексирание на дърво с N върха е $\leq O(N + \frac{N}{B} \log_2 N + 2^{2B} \times B^2)$.

Търсим B във вида $B = \frac{\log_2 N}{c_0}$. Тогава

$$O\left(N + c_0 N + 2^{\frac{2 \log_2 N}{c_0}} \left(\frac{\log_2 N}{c_0}\right)^2\right) = O\left(N + c_0 N + N^{\frac{2}{c_0}} \times \left(\frac{\log_2 N}{c_0^2}\right)^2\right). \text{ Така ако}$$

изберем $c_0 = 3$ или $c_0 = 4$ ще получим съответно $O\left(4N + \frac{N^{\frac{2}{3}} \log_2^2 N}{9}\right)$ или съответно

$$O\left(5N + \frac{\sqrt{N} \log_2^2 N}{16}\right) \text{ и в двата случая получаваме сложност } O(N).$$



В клетка с номер $j = T_{start}[i] + k|T_{v_0}| + d$, с $k < |T_{v_0}|$, $d < |T_{v_0}|$.

Записан отговор на заявката за (k, d) за дървото T_{v_0} .

↓
номер при
обхождане в
дълбочина

v - микро-корен:

1. DFS

- 1-ва цел: да пресметнем $\overline{1 \circ T_j}$
- 2-ра цел: да попълним сегмент от масива за микро-върховете

2. Поглеждаме на позиция $\overline{1 \circ w(T_v)}$ в масива от индекси на таблица.

2.1. сл. $\overline{1 \circ w(T_v)}$ е попълнено. Тогава запомняме $\overline{1 \circ w(T_v)}$ за v и приключваме с T_v

2.2. сл. $\overline{1 \circ w(T_v)}$ не е попълнено. Тогава още веднъж обхождаме в дълбочина T_v (в абсолютно същия ред) и попълваме сегмента от масива за таблиците.

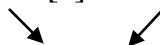
Вариант за наивен индекс:

$table[0 \dots |V| - 1][0 \dots |V| - 1]$

```
procedure DFS( $T, v, path, l$ ){  
    for  $j = 0$  to  $l$  do  
         $table[v][j] \leftarrow path[l - j]$   
    for  $j = l + 1$  to  $|V| - 1$  do  
         $table[v][j] = \perp$   
    for  $u : p(u) = v$  do  
         $path[l + 1] \leftarrow u$   
        DFS( $T, u, path, l + 1$ )
```

$table[v][j] \quad ?$

$k \leftarrow ind[v] - ind[root(v)]$


в масива от
микро-върхове

$d \leftarrow j$

$Tables[T_{start}[i] + k.size(root(v)) + j] \leftarrow ind[path[l - j] - ind[root(v)]]$