DAA Семинар 8

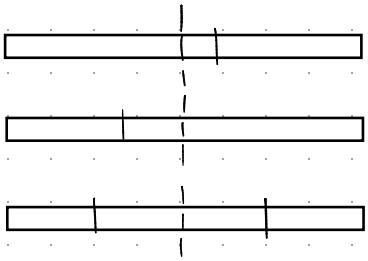
Зад. Даден е масив с числа $A[1..n]$, в който едно от тях се среща $\geq \frac{n}{2}$ на брой пъти. Кое е то? БОО. Нека $n \geq 1$.

Решение: I н. Да намерим мода чрез сортиране:

$$\underbrace{\Theta(n \lg n)}_{\text{сортиране}} + \underbrace{\Theta(n)}_{\text{линейно обхождане}} = \Theta(n \lg n)$$

II н. Сортиране и достъпване на средния елемент:

Възползваме се от това че знаем, че елементът се среща $\geq \frac{n}{2}$ пъти



$$\underbrace{\Theta(n \lg n)}_{\text{сорт.}} + \underbrace{\Theta(1)}_{\text{достъп}} = \Theta(n \lg n)$$

III н.

```
AlgMaj(A[1..n])
1  S ← празен стек
2  for i ← 1 to n
3      if S.isEmpty or S.top = A[i]
4          S.push(A[i])
5      else
6          S.pop()
7  return S.top
```

$T(n) = \Theta(n)$
$S(n) = O(n)$

IV н. "Симулиране" стека от III н. чрез брояч за големина размер и променлива за върха му

```
1 size ← 0, top ← 0 // може и начално и да е друго число
2 for i ← 1 to n
3     if size = 0 or (size ≠ 0 and top = A[i])
4         size ← size + 1
5         top ← A[i]
6     else
7         size ← size - 1
8 return top
```

$T(n) = \Theta(n)$
$S(n) = \Theta(1)$

Тр. Boyer-Moore

Семестриално 2018:

Зад. 4 Дадени са два масива $A[1,\ldots,n]$ и $B[1,\ldots,n+1]$. Масивът A е сортиран и съдържа само цели положителни числа. Масивът B се получава от A чрез вмъкване на точно една нула на някаква позиция $t \in \{1,\ldots,n\}$ в A и отместване с една позиция нагоре на елементите от A, които са от позиция t включително нататък. Ясно е, че B съдържа точно същите положителни числа като A и те са в точно същата наредба (сортирана), в каквато са в A. Ето пример за такива масиви:

$A = [2, 3, 6, 12, 13, 25, 47]$

$B = [2, 3, 6, 12, 0, 13, 25, 47]$

В примера, $n = 7$ и $t = 5$. Предложете колкото е възможно по-ефикасен алгоритъм, който има вход A и B и който връща t, тоест позицията, на която е вмъкната нулата. Обосновете кратко, но съдържателно коректността и сложността на предложения от Вас алгоритъм.

Решение: Ще решим задачата с двоично търсене

За $\forall k: 1 \le k \le n$:
- Ако $A[k] = B[k]$, то $k < t$
- Ако $A[k] < B[k]$, то $k > t$
- Ако $B[k] = 0$, то $t = k$

Псевдокод:

```
1  l ← 1, r ← n
2  while (l ≤ r)
3      mid ← ⌊(l+r)/2⌋
4      if B[mid] = 0
5          return mid
6      if A[mid] = B[mid]
7          l ← mid + 1
8      else
9          r ← mid - 1
10 return l
```

Коректността следва от горното твърдение.

Сложността е $\Theta(\lg n)$

Зад. Нека $A[1..n]$ и $B[1..m]$ са два сортирани масива, които използваме, за да представим множествата A и B.
Предложете алгоритъм, който намира:
а) $A \cap B$
б) $A \cup B$
в) $A \setminus B$
г) $A \Delta B$

Решение: а) $A \cap B$:

```
1.  i ← 1, j ← 1, k ← 1
2   while i ≤ n and j ≤ m
3       if A[i] = B[j]
4           output[k] ← A[i]
5           i ← i + 1
6           j ← i + 1
7           k ← k + 1
8       else if A[i] < B[j]
9           i ← i + 1
10      else
11          j ← j + 1
```

$T(n,m) = \Theta(n+m)$

$S(n,m) = \Theta(1)$

// За $A \setminus B$: Ако $A[i] = B[j] \Rightarrow i++; j++$

$<$ $\Rightarrow$ out $+A[i]$; $i++$

else $\Rightarrow$ $j++$

// За $A \cup B$ $= \Rightarrow$ out A; $i++$; $j++$

$<$ $\Rightarrow$ out A; $i++$

else $\Rightarrow$ out B; $j++$

+ добавяне на останалата от по-голямия масив

Зад. Намерете медианата на масива $C[1..2n]$, който се получава при сливането на $A[1..n]$ и $B[1..n]$, които са сортирани.

Решение: I н. Сливане на за $\Theta(n)$ в $\Theta(n)$ доп. памет и връщане $\frac{C[n] + C[n+1]}{2}$ за $\Theta(1)$

II н. Имитиране сливане без реално да го правим, като броим колко елемента сме взели и спираме при достигане нап.

$T(n) = \Theta(n)$ $S(n) = \Theta(1)$

III н. БОО тъкa медианата на A е по-малка от тази на B.

Никой от елементите в $A[1..\lfloor \frac{n}{2} \rfloor]$ няма да участва в намирането на медианата на обединения масив (ни може и на друг по-голям). Можем да ги игнорираме.

Аналогично, за B по-големите от медианата могат да бъдат „изхвърлени".

Така можем да приложим алгоритъма за по-малък вход.

Нека $m = 2n$ - размерът на входа. Тогава $T(m) = T(\frac{m}{2}+1)+1$

// $T(m) = T(\frac{m}{2}+2)$, ако $n = 2^k$ за $k \in \mathbb{N}$

Чрез Акра - Бази получаваме $T(n) = \Theta(\lg n)$; $S(n) = \Theta(1)$

Зад. Даден е $A[1..n]$. Да се намери k-тия елемент по големина в A.

Решение: I н. Сортиране + взимане на $A[n-k]$ $\rightarrow \Theta(n \lg n)$
или $A[k+1]$

II н. За линейно време чрез алгоритъма QuickSelect

- избира случаен pivot и търси в съотв. подмасив
- $T(n) = O(n \lg k)$ средно, но $O(n^2)$ в най-лошия случай

III PICK/Select/Median of medians // Има го в учебника

- намира медианите на (най-често) 5 подмасива, после медианата им и тя се използва за pivot
- $T(n) = \Theta(n)$

Зад (3SUM) Даден е $A[1..n]$ с уникални елементи. Да се намерят (ако има) $i \neq j \neq k \neq i$, за които $A[i] + A[j] + A[k] = 0$

Решение: I н. 3 вложени цикъла $\rightarrow \Theta(n^3)$

II н. Сортираме A и за всяка двойка индекси i, j търсим двоично дали $-(A[i] + A[j])$ присъства в масива (т.е. $-(A[i] + A[j]) = A[k]$ за някое k)

$$T(n) = \Theta(n \lg n) + \Theta(n^2 \lg n) = \Theta(n^2 \lg n)$$

// По принцип за търсенето може и да се ползва хеш. (слагаме елементите на A в хеша)

III. Чрез известен алгоритъм за решаване на задачата 3SUM:

```
1  Sort(A[1..n])
2  for i<-1 to n-2
3     j<-i+1, k<-n
4     while  j<k
5        if  A[i]+A[j]+A[k]>0
6           k<-k-1
7        else if  A[i]+A[j]+A[k]<0
8           j<-j+1
9        else
10          print(i,j,k)
11          j<-j+1
12          k<-k-1
```

$T(n) = \Theta(n \lg n) + \sum_{i=1}^{n} (n-(i+1)+1) =$

$= \Theta(n \lg n) + \sum_{i=1}^{n} (n-i) = O(n^2)$

// $T(n) = \Theta(n^2)$

// Alg3SUMCount в указания

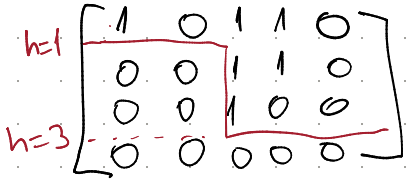
- Може задачата да е: $A[i]+A[j]+A[k] = c$, c - константа

→ Намаляваме ∀ елементи с $\frac{c}{3}$ и прилагаме горния 3SUM

Зад. Дадена е дулева матрица $A[1..n][1..m]$, $mn \geq 1$, чиито колони са сортирани низходящо.
Да се намери дължината на най-дългата колона от 1-ци.

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \rightarrow 3$$

Решение (идея):



Намираме височината на първата колона. Нататък разглеждаме другите колони от това ниво нататък.

$\Theta(n+m)$

Зад. Дадена е матрица $A[1..m][1..n]$ от числа, за които е вярно, че всяко е по-голямо от западния и северния си съсед.

По дадено число k да се намери дали то се съдържа в матрицата възможно най-ефикасно.

$$\begin{bmatrix} 1 & 2 & 7 & 12 & 14 \\ 3 & 6 & 8 & 16 & 17 \\ 5 & 10 & 11 & 20 & 21 \\ 13 & 15 & 19 & 22 & 25 \end{bmatrix} \quad k = 15$$

Идея: Започваме от северозточния край (или ЮЗ край) и ако $A[i][j]$ е по-малък от k, слизаме надолу, ако е по-голям → наляво

$\Theta(n+m)$