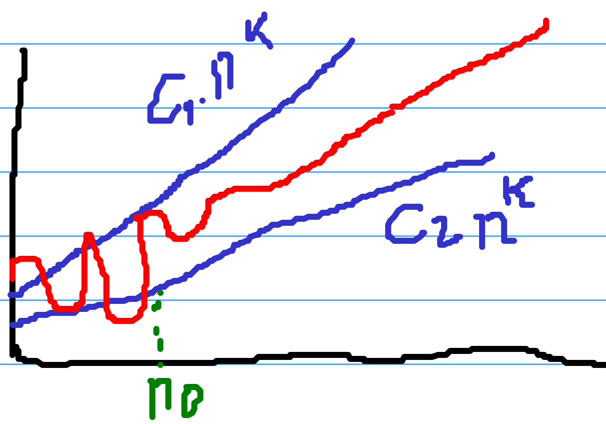
(5)

Зад 1

Нека k е фиксирано ест. число

Докажете, че при $n \geq k$ $\quad \binom{n}{k} \in \Theta(n^k)$

$$\Theta(n^k) = \{ g \mid \exists c_1 > 0 \; c_2 > 0 \; \exists n_0 \; \forall n \geq n_0 \quad c_1 \cdot n^k \leq g \leq c_2 n^k \}$$



$$c_1 n^k \leq \binom{n}{k} \leq c_2 n^k$$

- $k = 0$ $\qquad c_1 \cdot 1 \leq 1 \leq c_2 \cdot 1$ ✓
- $k \geq 1$ $\qquad \binom{n}{k} = \dfrac{n!}{(n-k)!\,k!} =$

$$= \frac{n\cdot(n-1)\cdot(n-2)\cdot(n-3)\dots\cancel{(n-k)}\cdot\cancel{(n-k-1)}\dots 1}{\cancel{(n-k)}\cancel{(n-k-1)}\cancel{(n-k-2)}\dots 1\cdot k!}$$

$$= \frac{n\cdot(n-1)\dots(n-k+1)}{k!} =$$

$$= \frac{n}{k}\cdot\frac{(n-1)}{(k-1)}\cdot\frac{(n-2)}{(k-2)}\dots\frac{(n-k+1)}{1}$$

?

$$C_1\cdot n^k \overset{?}{\leq} \frac{n}{k}\,\frac{(n-1)}{(k-1)}\,\frac{(n-2)}{(k-2)}\dots\frac{(n-k+1)}{1} \overset{?}{\leq} C_2\, n^k$$

$C_2 = 1$ ✓

$$C_1 \overset{?}{=} \frac{1}{k^k}$$

Ще покажем, че: $\frac{n^k}{k^k} \leq \frac{n}{k}\cdot\frac{(n-1)}{(k-1)}\dots\frac{(n-k+1)}{1}$

$$\frac{n}{k}\cdot\frac{n}{k}\cdot\frac{n}{k}\cdots \le \frac{n}{k}\cdot\overbrace{\frac{(n-1)}{(k-1)}}^{\boxed{\frac{n-m}{k-m}}}\overbrace{\frac{(n-2)}{(k-2)}}\cdots$$

- $k \le n \Rightarrow \frac{m}{n} \le \frac{m}{k}$

$$\Rightarrow 1 - \frac{m}{k} \le 1 - \frac{m}{n}$$

$$\Rightarrow \frac{k-m}{k} \le \frac{n-m}{n}$$

$$\Rightarrow \boxed{\frac{n}{k} \le \frac{n-m}{k-m}} \quad (*)$$

$$\frac{n}{k}\cdot\frac{n}{k}\cdot\frac{n}{k}\cdots\frac{n}{k} \le \frac{n}{k}\cdot\frac{(n-1)}{(k-1)}\cdot\frac{(n-2)}{(k-2)}\cdots\frac{n-k}{1}$$

$$\frac{n}{k} \le \frac{n}{k} \qquad \frac{n}{k} \overset{(*)}{\le} \frac{n-1}{k-1} \qquad \frac{n}{k} \overset{(*)}{\le} \frac{n-2}{k-2} \quad \ldots$$

$\Rightarrow$ Неравенството е изпълнено!

~~Зад 2~~

Сложността на generateAllSubsets ?

- Генерираме всички булеви вектори с дължина n.

$\{1,2,3,4,5\}$

$0\,0\,0\,0\,0 - \emptyset$

$0\,1\,0\,1\,1 - \{2,4,5\}$

$1\,0\,0\,1\,0 - \{1,4\}$

$\vdots$

Теоретична долна граница: 2^n

$$2^n \left\{ \begin{array}{l} \overbrace{0\,0 \ldots 0\,0\,0}^{n} \\ 0\,0 \ldots 0\,0\,1 \\ 0\,0 \ldots 0\,1\,0 \\ 0\,0 \ldots 0\,1\,1 \\ 0\,0 \ldots 1\,0\,0 \\ \quad \vdots \end{array} \right.$$

next

next

next

next

$\underline{\;0/1\;}\; 01\ldots111 \xrightarrow{next} \underline{\;0/1\;}\; 10\ldots000$

flip

Сложност на next в най-лош сл.

$0\underbrace{11111\ldots1}_{n-1} \xrightarrow{next} 1\underbrace{000\ldots0}_{n-1} \qquad \Theta(n)$

Амортизиран анализ на gAS

Разглеждаме: → while(__)

- false $\quad 2^n$ (обектът е готов)
- true $\quad i \quad \underbrace{\underset{1}{0/1}\;\underset{2}{0/1}\;\underset{3}{0/1}\ldots\underset{i}{1}\;\underset{i+1}{1}\;\underset{i+2}{1}\ldots\underset{n}{1}}_{2^{i-1}}$

$$\underbrace{2^n}_{} + \underbrace{\sum_{i=1}^{n} 2^{i-1}}_{} =$$

колко път връща false

колко пъти връща true

$$= \sum_{i=0}^{n} 2^n = 2^{n+1} - 1$$

Средно на обект: $\dfrac{2^{n+1}-1 \text{ (общо стъпки)}}{2^n \text{ (брой обекти)}} < 2 \quad \theta(1)$

next $= \theta(1)$ Амортизирано!

$\Rightarrow$ Сложността на gAS $\theta(2^n)$

зад 3 Сложност на
genAllSubsets-recursive

$$\text{gAS-rec}(n) \begin{cases} \to 1 \underbrace{\qquad\qquad\qquad}_{\text{gAS-rec}(n-1)} \\ \to 0 \underbrace{\qquad\qquad\qquad}_{\text{gAS-rec}(n-1)} \end{cases}$$

$T(n) = T(n-1) + T(n-1) + 1$

$T(n) = 2T(n-1) + 1$

метод хар. ур-е.

$\downarrow$

$\theta(2^n)$

Коректност на итеративни алгоритми.

1) Финитност - ще приключи ли?

2) Коректност на изхода

С $\underbrace{\text{инварианта на цикъла}}_{\text{предикат}}$

Предикатът трябва:

1) Да е верен при първото влизане в цикъла

2) Верността му да се запазва след всяко изпълнение.

3) От неговата вярност да следва верността на алгоритъма

Доказателство с индукция:

База – база

Инд. пр.
Инд. ст. } поддръжка

Терминация

последното изпълнение на въпросния ред

Зад4 Докажете, че ф-ята evenCount връща броя на четните числа в масива arr

Инварианта:

За всяка проверка за край на цикъла е изпълнено: evenCount **е броят четни числа в подмасива** arr[0..i-1].

База:

При първата проверка за край: i = 0, evenCount = 0;

Вярно ли е, че evenCount е броят на четните числа в подмасива arr[0...-1]?

arr[0...-1] - празен подмасив.

Броят на четните числа в празния подмасив е 0. **ОК!**

Поддръжка:

Допускаме, че инвариантата е изпълнена за някоя проверка за край на цикъла, **КОЯТО НЕ Е ПОСЛЕДНА!**

Допуснали сме, че evenCount е броят четни числа в подмасива arr[0..i-1].

На 9-ти ред правим проверка дали arr[i] е четно.

1сл. Проверката връща истина. Тогава arr[i] е **четно**.
Влизаме в тялото на if-а и инкрементираме evenCount.
evenCount' - новата стойност на evenCount (evenCount' = evenCount + 1).
От допускането: evenCount е броят четни числа в подмасива arr[0..i-1]
Но arr[i] е четно. Тогава броят на четните числа в arr[0..i] е evenCount + 1.
Тогава evenCount' е броят на четните числа в arr[0..i].
След това i се инкрементира и evenCount' става **броя на четните числа** в arr[0..i-1]. **ОК!**

2сл. Проверката връща лъжа.

Тогава arr[i] е **НЕЧЕТНО**. Т.е. броят на четните числа в подмасива arr[0...i-1] **е точно колкото е и в** в arr[0...i]. От предположението в променливата evenCount пазим броя на четните числа в подмасива arr[0...i-1]. Но тогава в evenCount **е броят на четните числа и в подмасива** arr[0...i].

Но след това i **се инкрементира** и evenCount е **броят на четните числа в подмасива** arr[0...i-1]. **ОК!**

Терминация:

Последната проверка за край на цикъла: i = len.

От инвиарантата: evenCount е броят на четните числа в подмасива arr[0...len-1].

Но това е целият масив. На ред 12 връщаме evenCount. **ОК!**

зад 2 taskMystery(x, y)

изход + коректност

1) $tm(x,y) = \begin{cases} x^y & x \neq 0 \vee y \neq 0 \\ 1 & else \end{cases}$

2) коректност

инварианта?

taskMystery(x,y)

При: 3^5

z	t	p
3	5	1
3	4	3
9	2	3
81	1	3
81	0	243

Връща p

1 сл x != 0 или y != 0

Инварианта: За всяка проверка за край на цикъла е изпълнено: **x^y = z^t * p**

База: При първото влизане в цикъла:

z = x

t = y

p =1

z^t * p = x^y * 1 = x^y. **OK!**

Поддръжка: Допускаме, че инвариантата е изпълнена за някоя проверка за край на цикъла, **която не е последна**.

Тогава t > 0.

1 сл. t е четно. Проверката връща истина.

z' (**новата стойност** на z) **z'**= z * z

t' (**новата стойност** на t) **t'**= t / 2

Вярно ли е: **z'** ^ **t'** * p = x ^ y ?

z' ^ **t'** * p = (z * z) ^ (t / 2) * p = z ^ (t / 2) * z ^ (t / 2) * p = z ^ t * p

= (от **допускането**) x^y. **OK!**

2 сл. t е нечетно. Проверката връща лъжа.

p' (**новата стойност** на p) **p'** = p * z

t' (**новата стойност** на t) **t'** = t – 1

Вярно ли е: z ^ **t'** * **p'** = x ^ y ?

Z ^ **t'** * **p'** = z ^ (t - 1) * z * p = z ^ t * p = (от допускането) x^y **OK**!

Терминация:

При последната проверка за край: t = 0.

=> x ^ y = z ^ t * p = z ^ 0 * p = p

На ред 37 връщаме точно p. **OK**!

<u>2сл. x = 0 и y = 0.</u>

Тогава t = 0. Тогава проверката на цикъла връща лъжа.

На ред 37 връщаме p, която има стойност 1-ца. OK!