Студент: Тимур Хабибуллин
Группа: МОАД
Дата: 5 мая 2022 г.

---

1. (a) $f(n) = O(g(n)) \overset{?}{\Leftrightarrow} \exists C > 0 : \forall n \in \mathbb{N} : f(n) \leqslant Cg(n)$:

   В случае $\Leftarrow$ очевидно: возьмем $N = 1$, добавим в определение $\forall n \geqslant N$ и получим определение O-большого.
   В случае $\Rightarrow$: пусть при данных C, N и $n = N - 1$ у нас нарушается неравество: $f(N-1) > Cg(N-1)$ (вообще говоря, оно может нарушаться и при другом n, но для определенности пусть будет так, или не нарушаться совсем, тогда нам подходит константа из определения O, но это не интересный случай). Поскольку $f(N-1), g(N-1) < \infty$, то введем $\frac{f(N-1)}{g(N-1)} = C_{N-1} > C$ и $C_{N-1} < \infty$. Аналогично можно ввести $C_{N-2} = \frac{f(N-2)}{g(N-2)}$ и т.д. Тогда мы получим набор $\{C_i\}_{i \in [1, N-1]}$, в котором обязательно есть $C_i > C$. Возьмем максимальное $C_i = C_{max}$ и тогда для всех $n \in \mathbb{N}$, в том числе и для $n < N$, будет выполнено неравенство $\frac{f(n)}{g(n)} \leqslant C_{max}$. Таким образом, мы получили определение без условия на N, т.е. они эквивалентны.
   Для случая $f, g : \mathbb{N} \to \mathbb{R}_{>0}$ аналогично.

   (b) $f(n) = o(g(n)) \overset{?}{\Leftrightarrow} \forall C > 0, \forall n \in \mathbb{N} : f(n) < Cg(n)$:

   Приведем пример, когда это не выполняется. Пусть $f(n) = 10n, g(n) = n^2$. Очевидно, что $f(n) = o(g(n)) : \forall C\ \exists N : \forall n \geqslant N\ 10n < Cn^2$ (например, $C = 1, N = 11$). Однако если убрать условие на $\exists N$, то неравенство не будет выполняться при $n \leqslant N$, а значит, эти определения не эквивалентны.
   Для случая $f, g : \mathbb{N} \to \mathbb{R}_{>0}$ аналогично.

2. Заполнить табличку

N	$f(n)$	$g(n)$	O	o	Θ	ω	Ω
a	n	n^2	+	+	-	-	-
b	$log^k n$	n^ε	+	+	-	-	-
c	n^k	c^n	+	+	-	-	-
d	$\sqrt{n}$	n^{sinn}	-	-	-	-	-
e	2^n	$2^{n/2}$	-	-	-	+	+
f	n^{logm}	m^{logn}	+	-	+	-	+
g	$log(n!)$	$log(n^n)$	+	-	+	-	+

   (a) Дано и очевидно

   (b) $f(n) = log^k n$, $g = n^\varepsilon$, где $\varepsilon > 0$
   $f = O(g)$ и $f = o(g)$, т.к. по правилу Лопиталя

$$\frac{f(n)}{g(n)} = \frac{log^k n}{n^\varepsilon} \sim \frac{klog^{k-1} n}{\varepsilon ln(2) n^\varepsilon} \sim ... \sim O(\frac{1}{n^\varepsilon}) \to 0 \leqslant C$$

   при $n \to \infty$, где $C > 0$ - любая наперед заданная константа.
   По этой же причине $f \neq \Omega(g)$, т.к. нельзя найти $C, N : \forall n \geqslant N : \frac{log^k n}{n^\varepsilon} \sim \frac{1}{n^\varepsilon} > C > 0$ и тем

более это не верно при условии $\forall C > 0$ - мы всегда сможем приблизиться к нулю сильнее, чем любая заданная наперед C.
Учитывая $f \neq \Omega(g)$, получаем $f \neq \Theta(g)$, т.к. $f = \Theta(g) \Leftrightarrow f = O(g) \cap f = \Omega(g)$

(c) $f(n) = n^k$, $g(n) = c^n = e^{nln(c)}$.
$f = O(g)$ и $f = o(g)$, т.к. по правилу Лопиталя

$$\frac{f(n)}{g(n)} = \frac{n^k}{e^{nln(c)}} \sim \frac{kn^{k-1}}{ln(c)e^{nln(c)}} \sim ... \sim O(\frac{1}{e^{nln(c)}}) \to 0 < const$$

при $n \to \infty$
Как и в предыдущем пункте, $f \neq \Omega(g)$, $f \neq \omega(g)$ и $f \neq \Theta(g)$.

(d) $f(n) = \sqrt{n}$, $g(n) = n^{sinn}$.
$f \neq O(g)$, т.к. нельзя найти $C, N : \forall n \geqslant N : \sqrt{n} < Cn^{sinn}$. При $n \to \infty$ можно всегда подобрать такие n и ε: $2\pi k - n < \varepsilon < 1/2, k \in \mathbb{N}$ (Например, $n = 50, \varepsilon = 0.16$). Тогда

$$n^{sinn} \approx (2\pi k)^{-\varepsilon + O(\varepsilon^3)} \quad \text{и} \quad \frac{f(n)}{g(n)} \approx= (2\pi k)^{1/2+\varepsilon+O(\varepsilon^3)} \to \infty.$$

Значит, не существует такой $C > 0$ и $N : \forall n \geqslant N : \frac{f(n)}{g(n)} \leqslant C$. Т.е. $f \neq O(g)$. Как следствие $f \neq o(g)$, поскольку какую бы мы не задали константу $C > 0$, всегда можно найти такие n, что $\frac{f(n)}{g(n)} \geqslant C$.

$f \neq \Omega(g)$, поскольку можем подобрать такие n и ε: $(\pi/2+2\pi k)-n < \varepsilon < 1/2, k \in \mathbb{N}$ (Например, $n = 1000, \varepsilon = 0.16$). Тогда

$$n^{sinn} \approx (\pi/2 + 2\pi k)^{(1-\varepsilon^2/2)} \quad \text{и} \quad \frac{f(n)}{g(n)} \approx (\pi k)^{-1/2+\varepsilon^2/2} \to 0.$$

Получаем, что не существует такой $C > 0$ и $N : \forall n > N\ \frac{f(n)}{g(n)} \geqslant C$, т.е. $f \neq \Omega(g)$, и по той же причине $f \neq \omega(g)$. Как следствие, $f \neq \Theta(g)$

(e) $f(n) = 2^n, g(n) = 2^{n/2}$
Сразу видно, что $f(n) \neq O(g(n))$, т.к. $\frac{f(n)}{g(n)} = 2^{n/2} \to \infty$ при $n \to \infty$. Т.е. $\nexists C, N : \forall n \geqslant N$ $\frac{f(n)}{g(n)} < C$. Следовательно, $f(n) \neq o(g(n))$.
$f(n) = \omega(g(n))$, т.к. $\forall C > 0\ \exists N : \forall n > N\ \frac{f(n)}{g(n)} = 2^{n/2} > C$. Как следствие $f(n) = \Omega(g(n))$, т.к. раз это верно $\forall C > 0$, то это значит, что $\exists C > 0$.
$f(n) \neq \Theta(g(n))$, поскольку $f(n) \neq O(g(n))$.

(f) $f(n) = n^{logm}, g(n) = m^{logn}$
Так как $log(x) = \frac{lnx}{ln2}$, то $f(n) = exp(lnn \cdot lnm/ln2) = g(n)$. Т.е. получаем, что $f(n) = g(n)$.

$f(n) = O(g(n))$, так как при $C \geqslant 1\ \forall n \in \mathbb{N}\ f(n) \leqslant Cg(n)$. Однако $f(n) \neq o(g(n))$, т.к. неравенство $f(n) < Cg(n)$ не выполняется при $C < 1$
$f(n) = \Omega(g(n))$, т.к. $\forall n \in \mathbb{N}\ f(n) \geqslant Cg(n)$ при $C \leqslant 1$. Следовательно, $f(n) \neq \omega(g(n))$.
Поскольку $f(n) = O(g(n))$ и $f(n) = \Omega(g(n))$, то $f(n) = \Theta(g(n))$.

(g) $f(n) = log(n!)$, $g(n) = log(n^n) = nlog(n)$
Формула Стирлинга: $f(n) = nlog(n) - n + O(log(n))$.

$f(n) = O(g(n))$, так как $\frac{f(n)}{g(n)} = \frac{nlog(n) - n + O(n)}{nlog(n)} = 1 - \frac{1}{log(n)} + O(\frac{1}{n}) < 1$. Получается, что для $\forall C \geqslant 1$ $\frac{f(n)}{g(n)} \leqslant C$ при $N \geqslant 2$. $f(n) \neq o(g(n))$, поскольку нам не подходят $C < 1$.
$f(n) = \Omega(g(n))$, т.к. $\forall n \geqslant 2$ $\frac{f(n)}{g(n)} \geqslant C = \frac{1}{2}$. Действительно, из формулы для $\frac{f(n)}{g(n)}$ видно, что с ростом n оно монотонно увеличивается и стремится к 1. Значит, минимум достигается при $n = 2$, а тогда $\frac{log(n!)}{log(n^n)} \geqslant \frac{1}{2}$. Из этого следует, что неравенство $\frac{f(n)}{g(n)} > C$ нарушается при $C < \frac{1}{2}$, значит, $f(n) \neq \omega(g(n))$.
Поскольку $f(n) = O(g(n))$ и $f(n) = \Omega(g(n))$, то $f(n) = \Theta(g(n))$.

3. Задача про массив и m операций add(x, l, r):

   Пусть исходный массив считан в массив $a[i], i = 1...n$.

   (a) Создадим за $O(n)$ массив из нулей $b[i], i = 1...n$ и для каждого запроса $add(x, l, r)$ изменим $b[i]$ следующим образом:
   $$b[l] += x$$
   $$b[r+1] += -x, \text{если } r < n$$
   Это займет время $O(m)$.

   (b) Затем за $O(n)$ сделаем еще одно преобразование $b[i]$:
   $$b[i+1] += b[i], i = 1...n-1$$
   В массиве начиная с каждого $b[r]$ в следующую позицию будет добавляться свой x, и дойдя до $b[r+1]$ этот x удалится.

   (c) За время $O(n)$ сложим исходный массив $a[i]$ с $b[i]$ и выведем результат.

   Итоговая сложность алгоритма $O(n) + O(m) = O(m + n)$

4. $T(n) = 2T(\lfloor log(n) \rfloor) + 2^{log^*(n)}$, где $log^*(n)$ - итерированный логарифм.

   Пусть $log^*(n) = k$. Поскольку итерированный логарифм равен числу итерированных логарифмирований аргумента n, необходимых для того, чтобы результат стал не больше 1, то если подставить в качестве аргумента $\lfloor log(n) \rfloor$, то значение итерированного логарифма уменьшится на 1:
   $$log^*(\lfloor log(n) \rfloor) = k - 1$$
   Как видно из рекурентного соотношения, на первом шаге выполняется задача размера n и тратится $2^{log^*(n)} = 2^k$ времени. На следующем шаге мы получаем задачу размера $\lfloor log(n) \rfloor$ и тратим $2 \cdot 2^{log^*(\lfloor log(n) \rfloor)} = 2 \cdot 2^{k-1} = 2^k = 2^{log^*(\lfloor log(n) \rfloor)}$ времени. И так далее на каждом шаге будем тратить $2^{log^*(\lfloor log(n) \rfloor)}$ времени. Общее время работы - сумма потраченного времени на каждом шаге. Глубина дерева рекурсии есть $k = log^*(n)$.
   В результате получаем $T(n) = O(k * 2^k) = O(log^*(n) \cdot 2^{log^*(n)})$