Complexité

I. Domination

Définition

Soient $(u_n)_{n\in\mathbb{N}}$ et $(v_n)_{n\in\mathbb{N}}$ deux suites réelles. On dit que (u_n) est **dominée** par (v_n) s'il existe un $n_0 \in \mathbb{N}$ et une constance C telle que pour tout $n \geqslant n_0$ on a $|u_n| \leqslant C|v_n|$.

On écrit alors $u_n = O(v_n)$ et on dit que u_n est un "grand O" de v_n.

Proposition

Soient $(u_n)_{n\in\mathbb{N}}$ et $(v_n)_{n\in\mathbb{N}}$ deux suites réelles et supposons que v_n est non nul pour n assez grand.

Alors $u_n = O(v_n)$ si et seulement si la suite $\dfrac{u_n}{v_n}$ est bornée

par exemple si la suite possède une limite finie

- $n = O(n^2)$ car $\lim_{n\to+\infty} \frac{n}{n^2} = 0$
- $n = O(3n+1)$ car $\lim_{n\to+\infty} \frac{n}{3n+1} = \frac{1}{3}$
- $\ln(n) = O(n)$ car $\lim_{n\to+\infty} \frac{\ln(n)}{n} = 0$ (T.C.C.)
- $n = O(e^n)$ car $\lim_{n\to+\infty} \frac{n}{e^n} = 0$ (T.C.C.)
- $e^n \neq O(n)$ car $\lim_{n\to+\infty} \frac{e^n}{n} = +\infty$ (T.C.C.)

- si $u_n = O(w_n)$ et $v_n = O(w_n)$ alors $u_n + v_n = O(w_n)$
- si $u_n = O(w_n)$ et que $\lambda \in \mathbb{R}$ alors $\lambda u_n = O(w_n)$.
- su $u_n = O(w_n)$ et que v_n est bornée alors $u_n v_n = O(w_n)$
- si $u_n = O(w_n)$ et $v_n = O(z_n)$ alors $u_n v_n = O(w_n z_n)$

Attention

On a $n^{-1} = O(1)$ mais $\ln(n) \neq O(\ln(0))$
En général, il faut faire attention quand on applique des fonctions à l'intérieur des O.

II. Complexité

Entrée : L, x
Tant que $i < n$ **faire :**
si $L[i] = x$ **alors sortir**
sinon $i \leftarrow i + 1$
Sortie : i

Attention

- pas de formule précise pour le temps de calcul $T(n)$ (dépend de x, de l'ordre dans la liste, ...) $\Rightarrow$ **majoration**
- pas de formule pour le temps que prend une comparaison (dépend du type d'objet, de l'ordinateur, ...) $\Rightarrow$ **majoration à une constante multiplicative près**

temps de calcul $T(n)$ est donc au plus de n comparaisons

$$\Rightarrow T(n) = O(n)$$

Entrée : (a, b)
tant que $b \neq 0$ **faire :**

$$a = bq + r$$
$$a \leftarrow b,\ b \leftarrow r$$

fin tant que
Sortie : a

On note $T(a, b)$ le temps de calcul (numbre d'opérations élémentaires) du $\mathrm{pgcd}(a, b)$ par l'algorithme d'Euclide

Principe de la majoration de $T(a, b)$:

- on démontre que le cas le pire est le cas de la suite de Fibonacci
- on fait le calcul dans le cas de la suite de Fibonacci

Définition

Suite de Fibonacci $F_0 = 0$, $F_1 = 1$ et pour tout $n \geqslant 2$

$$F_n = F_{n-1} + F_{n-2}$$

Théorème

$a, b \in \mathbb{N}$, $a > b$, si $T(a, b) \geqslant n$ alors $a \geqslant F_{n+2}$ et $b \geqslant F_{n+1}$

La démonstration de fait par récurrence

Initialisation : Si $T(a,b) \geqslant 1$, alors $b \neq 0$, donc $b \geqslant 1 = F_2$ et $a > b$ donc $a \geqslant 2 = F_3$.

Hérédité : Supposons le résultat vrai au rang n.

$$a = bq + r \text{ avec } 0 \leqslant r < b$$

$$T(a,b) = T(b,r) + 1 \geqslant n \Rightarrow T(b,r) \geqslant n - 1$$

$$\text{(HR)} \Rightarrow b \geqslant F_{n+1} \text{ et } r \geqslant F_n$$

$$a = bq + r \geqslant qF_{n+1} + F_n \geqslant F_{n+1} + F_n = F_{n+2}$$

CQFD

Théorème

Pour tout $n \in \mathbb{N}$, $F_n = \frac{1}{\sqrt{5}}\left(\varphi^n - \varphi^{-n}\right)$ avec $\varphi = \frac{1+\sqrt{5}}{2}$

La démonstration se fait par récurrence

$$c \geqslant F_n \Leftrightarrow c \geqslant \frac{1}{\sqrt{5}}\left(\varphi^n - \varphi^{-n}\right)$$

Comme $\varphi^{-1} \approx 0.61 < 1$ et que c est entier

$$\begin{aligned} c \geqslant F_n &\Leftrightarrow c+1 \geqslant \frac{1}{\sqrt{5}}\varphi^n \\ &\Leftrightarrow \sqrt{5}(c+1) \geqslant \varphi^n \\ &\Leftrightarrow \ln(\sqrt{5}(c+1)) \geqslant n\ln(\varphi) \\ &\Leftrightarrow \log_\varphi(\sqrt{5}(c+1)) \geqslant n \end{aligned}$$

Théorème

$$T(a,b) \leqslant \log_\varphi\left(\sqrt{5}\max(a,b)\right)$$

Comparaison	Complexité	Problème exemple
$O(1)$	constante	Accès tableaux
$O(\log(n))$	logarithmique	Dichotomie
$O(n)$	linéaire	Parcours d'une liste de longueur n
$O(n!)$	factorielle	Problème du voyageur de commerce pour n villes
$O(2^{2^n})$	doublement exponentielle	Décision de l'arithmétique de Presburger pour un énoncé de longueur n
$O(n \log(\log n))$		Crible d'Eratosthène donnant tous les nombres premiers $\leqslant n$.

Attention

- Les problèmes polynomiaux (ou moins) sont considérés comme simple
- Les problèmes exponentiels (ou plus) sont considérés comme complexe

Youtube : ScienceEtonnante
Nos algorithmes pourraient-ils être BEAUCOUP plus rapides ? (P=NP ?)