Časovna zahtevnost algoritma (neformalni opis)

time complexity

Želimo "grobo" oceniti "čas izvajanja" algoritma. Iskali bomo zgornjo mejo, torej oceno za čas izvajanja v najslabšem primeru. Čas izvajanja je odvisen od velikosti naloge, ki jo rešujemo.

$T : n \longmapsto$ najvećji čas izvajanja, ki ga algoritem porabi za rešitev naloge velikosti $\leq n$

(načeloma lahko ∞)

n ↑ mera za velikost naloge

Dejanski fizični čas izvajanja je seveda odvisen od strojne in programske opreme, ki jo uporabimo. Zato bomo ocenjevali le "red velikosti" časovne zahtevnosti, pri čemer bomo premislili, da konstante, ki so skrite v redu velikosti, niso astronomske.

Opomba.

- Poleg časovne včasih ocenjujemo tudi prostorsko zahtevnost postopkov (koliko pomnilnika potrebujejo za izvajanje ...)
- Če poznamo porazdelitev vhodnih podatkov, lahko ocenjujemo tudi pričakovano časovno zahtevnost.

Def.

$f, g : \mathbb{N} \longrightarrow \mathbb{R}_+$

morda za končno vrednosti nista definirani

$f = O(g)$ (tudi $f(n) = O(g(n))$, $f \in O(g)$)

$\Updownarrow$ def

$\exists C > 0 \; \exists N \in \mathbb{N} \; \forall n \geq N : f(n) \leq C \cdot g(n)$

"f je reda kvečjemu g"

Zgledi

$\log^2 n = O(n)$ velja za naravni, dvojiški, desetiški log...

$3n + 7 = O(n)$ $\Leftarrow$ tudi $\Theta(n)$

$n^{100} = O(2^{n/100})$

$\sum_{k=1}^{n} (100k^2 - 10^{10}k) = O(n^3)$ $\Leftarrow$ tudi $\Theta(n^3)$

Def.

$f = \Omega(g) \overset{def}{\Longleftrightarrow} g = O(f)$ "f je reda vsaj g"

$f = \Theta(g) \overset{def}{\Longleftrightarrow} f = O(g)$ in $f = \Omega(g)$ "f je reda g"

$f = o(g) \overset{def}{\Longleftrightarrow} \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$ "f je asimptotično zanemarljivo majhna v primerjavi z g"

$f = \omega(g) \Longleftrightarrow g = o(f)$

Pomembni oznaki sta O in Θ.

Tipične časovne zahtevnosti:

linearna $\Theta(n)$

- iskanje največjega elementa v tabeli,
- skalarni produkt dveh vektorjev

kvadratna $\Theta(n^2)$

- seštevanje matrik, transponiranje matrik, množenje matrike in vektorja (matrika je velikosti $n \times n$, vektor ima n komponent; za mero velikosti problema vzamemo n)
- urejanje z vstavljanjem

kubična $\Theta(n^3)$

- običajno množenje matrik
- pravilno sprogramirana madžarska metoda

polinomska $O(p(n))$, kjer je p polinom

eksponentna $\Theta(2^{p(n)})$, kjer je p polinom stopnje ≥ 1 s pozitivnim vodilnim koeficientom

čas. zaht.	povečanje velikosti	povečanje časa
n	krat 2	krat 2
n^2	krat 2	krat 4 $(=2^2)$
n^3	krat 2	krat 8 $(=2^3)$
2^n	za 1	krat 2 $(\Leftrightarrow 2^{n+1} = 2 \cdot 2^n)$