Cím

Princzes Barnabás

2023. március 17.

Kivonat

Lent hagytam a füzetem és jövő héten doga szóval time to kick ass! Írjunk egy jó kis jegyzetet.

1. Előadás

1.1. Alapok

Van a $\{B\}A\{K\}$ azaz a Bemeneti feltétel (egy logikai állítás) Algoritmus és Kimeneti feltétel (szintén egy logikai állítás).
Legyen A algoritmus. $e_1, \ldots, e_m$ elemi műveletek az algoritmusban és t_i az edott e_i-hez tartozó időigény. Az algoritmus tényleges futási ideje $T(A,x)$ ahol x egy bemenet és a bemenet mérete $|x|$ például (tömb, halmaz...) esetében az elemek száma. Az e_i és t_i és $|x|$ együtt a bonyolultság mértéke.

1.2. Esetek

- Legjobb eset:
$$T_{lj} = \min\{T(A,x) : |x| = n\}$$
- Legrosszabb eset:
$$T_{lr} = \max\{T(A,x) : |x| = n\}$$
- Átlagos eset: Legyen $\Pr(x)$ annak a valószínűsége, hogy épp x lesz A algoritmus bemenete, ekkor:
$$T_a(A,n) = \sum_{|x|=n} \Pr(x)T(A,x)$$

```
public static int Keres(int[] A, int x){
  int i=0;                            //c1
  while (i<A.length && A[i]!=x){      //c2
    i++;                              //c3
  }
  return i;                           //c4
}
```

Itt Keres(A,n,x) futási ideje: $c_1 + (i+1)c_2 + ic_3 + c_4$

$$T_{lj} = c_1 + c_2 + c_4 = O(1)$$

$$T_{lr} = c_1 + (n+1)c_2 + nc_3 + c_4 = (c_2 + c_3)n + c_1 + c_2 + c_4 = O(n)$$

Az átlagos eset számításához vezessünk be pár fogalmat:
Legyen B_0 a legjobb bemeneti eset amikor is egyből megtaláljuk a keresett elemet
Legyen B_n amikor nem találjuk meg.
Minden lehetséges bemenet $\in B_i \mid i = 0, \ldots, n-1$
Legyen D az Integer összes lehetséges értéke (nagy szám).
Annak hogy az keresendő számot pont kiszúrjuk a valószínűsége: $p = \frac{1}{D}$
Annak, hogy egy olyat találunk amelyik nem a keresett szám: $q = 1 - \frac{1}{D}$
$Pr((A, n, x) \in B_i) = q^i p, i = 0, \ldots, n-1$
Itt q^i jelöli, hogy hányszor nem találtuk meg és p amikor megtaláltuk. Az utolsó lehetséges esetnél viszont csak q eset fordul elő az pedig n-szer ugyanis végigmegyünk és nem találjuk meg a keresett elemet. $Pr((A, n, x) \in B_n) = q^n$,

$$T_a(n) = \sum_{i=0}^{n} Pr((A, n, x) \in B_i)(c1 + (i+1)c2 + ic3 + c4)$$

Az átlagos esetet úgy kapjuk meg, ha minden lehetséges bemenet futási idejét megszorozzuk az adott bemenet valószínűségével és az eseteket összeadjuk.
A B_n bemenetre vonatkozó eset specifikus így emeljük ki a többi közül:

$$T_a(n) = q^n(c1 + (n+1)c2 + nc3 + c4) + \sum_{i=0}^{n-1} q^i p(c1 + (i+1)c2 + ic3 + c4)$$

Mivel $q^n \to 0$ (0-hoz tart) ezért felfele kerekítjük 1-re. Mivel a Σ alatt mindent beszorzunk $p(\ldots)$-al ahol a zárojelben $i < n$ így azt ki is emelhetjük ugyanúgy felfele kerekítve i-t és $i+1$-et kicserélve n-re.

$$T_a(n) \lessapprox \hat{T_a(n)} = (c1 + (i+1)c2 + ic3 + c4) + (c_1 + n(c_2 + c_3) + c_4)p \sum_{i=0}^{n-1} q^i$$

A Σ tag egyenlő a mértani sorozat összegképletével, behelyettesítjük:

$$\hat{T_a(n)} = (c1 + (i+1)c2 + ic3 + c4) + (c_1 + n(c_2 + c_3) + c_4)p \frac{q^n - 1}{q - 1}$$

Mivel $p = 1 - q$ ezért átalakítunk:

$$\hat{T_a(n)} = (c1 + (i+1)c2 + ic3 + c4) + (c_1 + n(c_2 + c_3) + c_4)(1 - q^n)$$

Mivel $(1 - q^n) \to 1$ megint felfele kerekítünk, majd összevonunk.

$$\hat{T_a(n)} \lessapprox \hat{\hat{T_a(n)}} = (2c_2 + 2c_3)n + 2c_1 + c_2 + 2c_4$$

Itt pedig n a domináns tag szóval:

$$T_a(n) = O(n)$$

1.3. Praktikus képletek

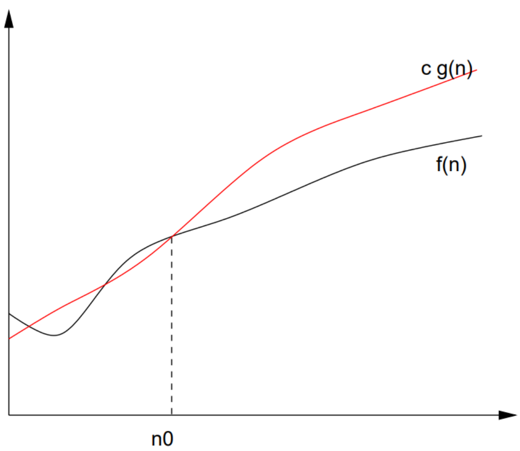
$$\begin{aligned}
T_{lj} =& \min\{T(A,x) : |x| = n\} \\
T_{lr} =& \max\{T(A,x) : |x| = n\} \\
T_a(A,n) =& \sum_{|x|=n} \Pr(x)T(A,x) \\
\sum_{i=1}^{n} i =& \frac{n(n+1)}{2} \quad (a_1 = 1,\ d = 1 \text{ speciális számtani sorozat összegképlete}) \\
\sum_{i=1}^{n} i^2 =& \frac{n(n+1)(2n+1)}{6} \\
\sum_{i=1}^{n} n =& n^2 \\
\sum_{i=m}^{n} q^i =& \frac{q^m - q^{n+1}}{1-q} \quad (\text{mértani sorozat összegképlete})
\end{aligned}$$

2. Függvények növekedési rendje

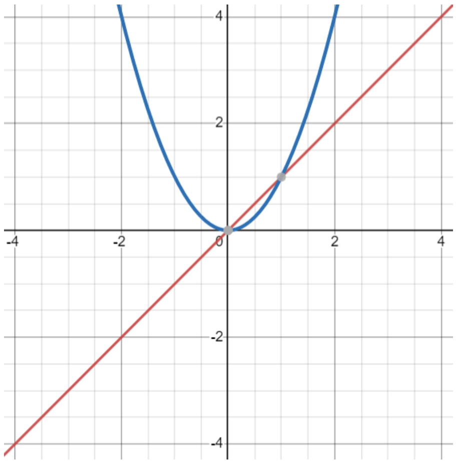
2.1. O - Nagy ordó

$$O(g(n)) = \{f(n) : (\exists c > 0, n_0 \geq 0)(\forall n \geq n_0)(0 \leq f(n) \leq cg(n))\}$$

Nagy ordó egy halmaz amelyekben adott függvények vannak, jelöljünk egy ilyen függvényt $f(n)$-el. $f(n)$ függvény ahol létezik egy c konstans úgy, hogy lesz egy n_0 küszöbszám amire igaz, hogy minden n_0-nál nagyobb n-számra $f(n)$ nem negatív és $cg(n)$ alatt van.
$f(n) \in O(g(n))$ helyett szoktuk a $f(n) = O(g(n))$ jelölést alkalmazni.



$f(n) = O(g(n))$ mivel $g(n)$-re tudunk választani olyan pozitív konstansot, hogy mindig $f(n)$ fölött maradjon.



Hasonlítsuk össze x és x^2 függvényt

Nem tudunk olyan c konstanst választani, hogy azzal $c\,x$ egy n_0 érték fölött mindig nagyobb legyen mint x^2 ezért:

$O(x) \neq O(x^2)$

2.2. Ω - Omega

Ugyanaz mint a O csak ez alulról korlátoz.

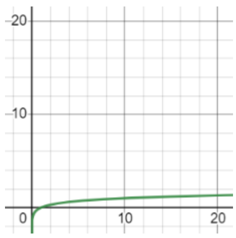
$$\Omega(g(n)) = \{f(n) : (\exists c > 0, n_0 \geq 0)(\forall n \geq n_0)(0 \leq cg(n) \leq f(n))\}$$

2.3. Θ - Theta

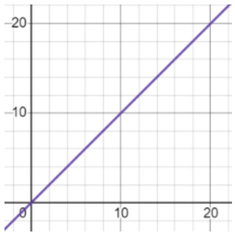
Ez alulról és felülről is korlátol. Azaz egy adott értéktől a legjobb és a legrosszabb eset is ugyanazt a nagyságrendet követi.

$$\Theta(g(n)) = \{f(n) : (\exists c_1, c_2 > 0, n_0 \geq 0)(\forall n \geq n_0)(0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n))\}$$

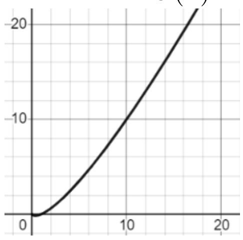
2.4. Nevezetes függvényosztályok



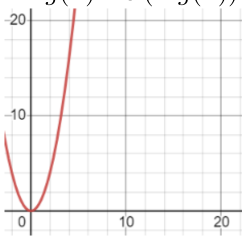
Logaritmikus - $O(lg(n))$



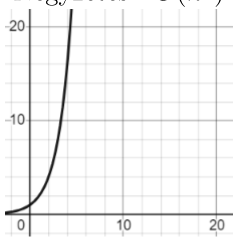
Lineáris - $O(n)$



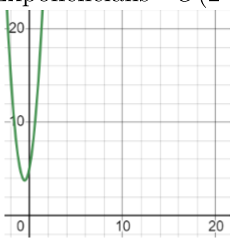
$nlog(n)$ - $O(nlg(n))$



Négyzetes - $O(n^2)$



Exponenciális - $O(2^x)$



$$\text{Polinomiális - } O(\sum_{i=0}^{k} a_i n^i)(a_k > 0)$$

Ez utóbbi a legösszetettebb így itt sokkal elővigyázatosabbnak kell lenni mikor azt nézzük, hogy melyiknél nagyobb.