ASIMPTOTSKO PONAŠANJE

Strukture podataka i algoritmi 1

Oznake

- Asimptotska gornja granica – O (veliko O)
- Asimptotska donja granica - Ω
- Još neke oznake – Θ i o
- Asimptotska analiza nekih algoritama

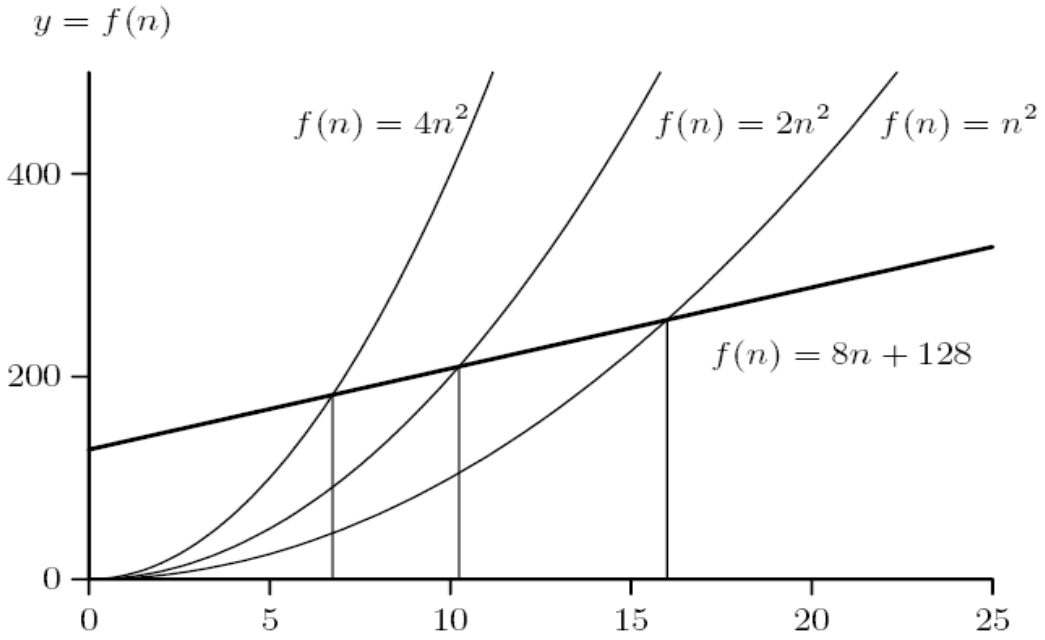
Veliko O - definicija

- Asimptotska gornja granica O
- Posmatrajmo funkciju $f(n)$ koja je ne-negativna za sve prirodne brojeve n >= 0.
- Kažemo da je $f(n)=O(g(n))$ ako postoji prirodan broj n_o i konstanta $c > 0$ tako da za sve brojeve $n >= n_0$, važi $f(n) <= cg(n)$.

Primer

- Posmatrajmo funkciju $f(n)=8n+128$.
- $f(n)=O(n^2)$?

$$f(n) \le cn^2, c = 1$$
$$\Rightarrow 8n + 128 \le n^2$$
$$\Rightarrow 0 \le n^2 - 8n - 128$$
$$\Rightarrow 0 \le (n-16)(n+8)$$
$$\forall n \ge 0, (n+8) > 0$$
$$(n_0 - 16) \ge 0 \Rightarrow n_0 = 16$$



Veliko O zablude

- Zabluda 1: $f_1(n) = O(g(n)) \wedge f_2(n) = O(g(n)) \Rightarrow f_1(n) = f_2(n)$
  - Primer:

$$f_1(n) = n \wedge f_2(n) = n^2,\ f_1(n) = O(n^2) \wedge f_2(n) = O(n^2),\ f_1(n) \neq f_2(n)$$

- Zabluda 2: $f(n) = O(g(n)) \Rightarrow g(n) = O^{-1}(f(n))$
  - Nije jednakost u matematičkom smislu

Osobine velikog O

- Aditivnost

$$f_1(n) = O(g_1(n)) \wedge f_2(n) = O(g_2(n)) \Rightarrow f_1(n) + f_2(n) = O(\max(g_1(n), g_2(n))$$

- Aditivnost opet

$$(f(n) = f_1(n) + f_2(n))$$
$$\wedge (\forall n \geq 0 \Rightarrow f_1(n), f_2(n) > 0)$$
$$\wedge (\lim_{n \to \infty} f_2(n) / f_1(n) = L, L \geq 0) \Rightarrow$$
$$\Rightarrow f(n) = O(f_1(n))$$

- Proizvod

$$f_1(n) = O(g_1(n)) \wedge f_2(n) = O(g_2(n)) \Rightarrow f_1(n) \times f_2(n) = O((g_1(n) \times g_2(n))$$

- Proizvod opet

$$(f_1(n) = O(g_1(n))$$
$$\wedge (\forall n \geq 0 \Rightarrow g_2(n) > 0) \Rightarrow$$
$$\Rightarrow f_1(n) \times g_2(n) = O(g_1(n) \times g_2(n))$$

- Tranzitivnost

$$f(n) = O(g(n)) \wedge g(n) = O(h(n)) \Rightarrow f(n) = O(h(n))$$

Polinomi i algoritmi

- Za svaki polinom $f(n) = \sum_{i=0}^{m} a_i n^i = a_m n^m + a_{m-1} n^{m-1} + \ldots + a_1 n^1 + a_0$
- važi $f(n) = O(n^m)$

- Za logaritme važi $\forall k \in N, k \geq 1 \Rightarrow \log^k n = O(n)$

Tesno ograničenje

- Veliko O daje informaciju samo o gornjem ograničenju, bez predstave o tome koliko je funkcija bliska asimptotskom ponašanju.
- Tesno ograničenje: razmotrimo funkcije $f(n)=O(g(n))$. Ako za svaku funkciju $h(n)$ takvu da je $f(n)=O(h(n))$ i važi da je $g(n)=O(h(n))$ onda kažemo da je funkcija $g(n)$ tesno asimptotsko ograničenje za funkciju $f(n)$.

Još veliko O zablude

- Zabluda 3:

$$f(n), g_1(n), g_2(n) \geq 0,\ f(n) = g_1(n) \times g_2(n)$$
$$\forall n \geq 0,\ f(n) \leq c g_1(n)$$
$$\text{if } c = g_2(n) \Rightarrow f(n) = O(g_1(n))$$

  - c mora biti konstanta, ne funkcija od n

- Zabluda 4:

$$f_1(n), f_2(n), g_1(n), g_2(n) \geq 0,$$
$$f_1(n) = O(g_1(n)) \wedge f_2(n) = O(g_2(n))$$
$$\forall n \geq 0, g_1(n) < g_2(n) \Rightarrow$$
$$\exists n_0, f_1(n_0) < f_2(n_0)$$

- Zabluda 5:

$$f(n), g(n) \geq 0,$$
$$f(n) = O(g(n)) \text{ xor } g(n) = O(f(n))$$

Konvencije za pisanje

- Ukloniti sve osim značajnog terma
  - Primer $O(n^2 + n\log n + n) \rightarrow O(n^2)$
- Ukloniti konstante
  - Primer $O(3n^2) \rightarrow O(n^2)$

    $O(1024) \rightarrow O(1)$

Primer(1)

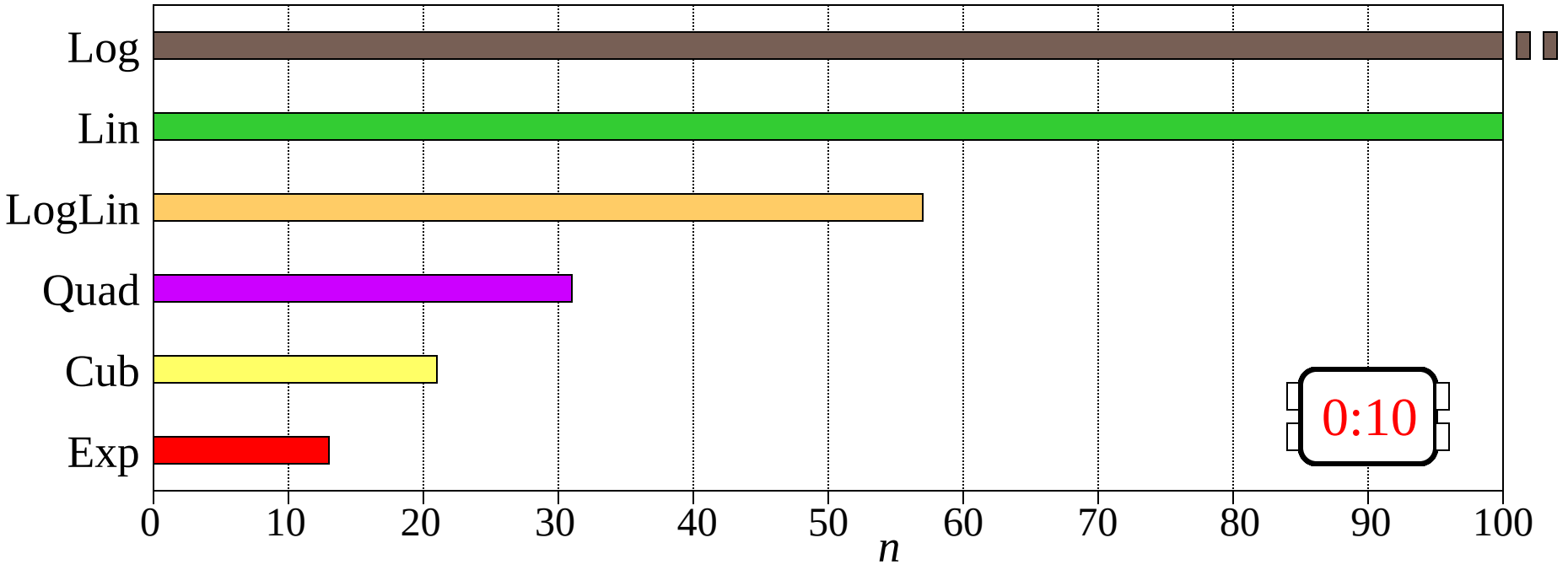
- Razmotrimo problem u kojem se procesira n podataka.
- Razmatramo nekoliko algoritama koji rešavaju ovaj isti problem. Pretpostavimo da su njihove vremenske složenosti date sledećim funkcijama:

Algoritam Log:	$0.3 \log_2 n$	sekundi
Algoritam Lin:	$0.1\ n$	sekundi
Algoritam LogLin:	$0.03\ n \log_2 n$	sekundi
Algoritam Quad:	$0.01\ n^2$	sekundi
Algoritam Cub:	$0.001\ n^3$	sekundi
Algoritam Exp:	$0.0001\ 2^n$	sekundi

- I grafički …

Primer (2)

- Uporedimo koliko podataka (*n*) svaki algoritam može da procesira u 1,2,...,10 sekundi:



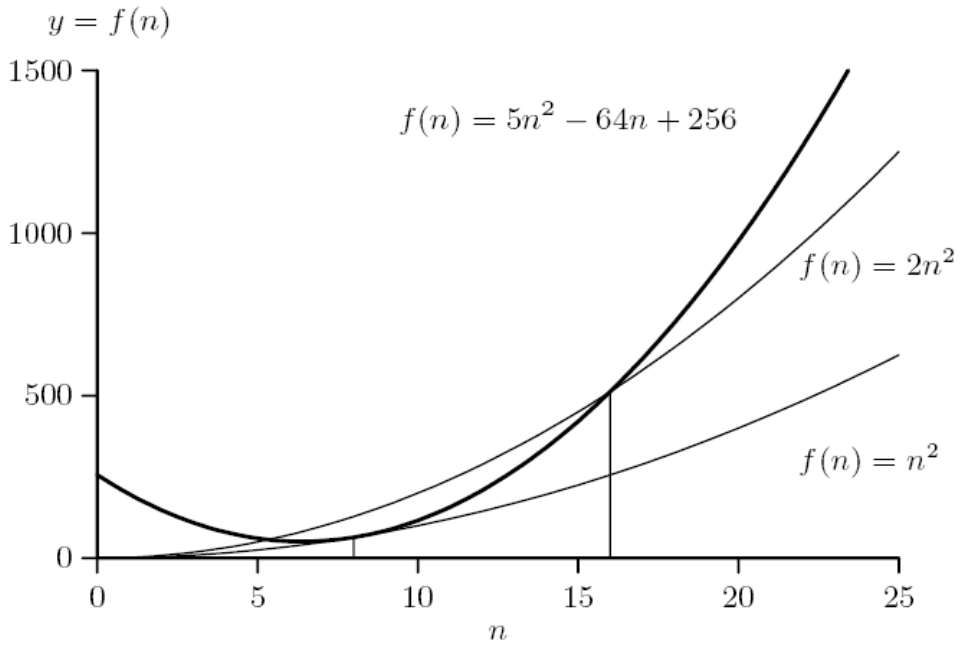
Ω definicija

- Asimptotsko donje ograničenje Ω
- Razmotrimo funkciju $f(n)$ koja je ne-negativna za sve prirodne brojeve n >= 0.
- Kažemo da je $f(n) = \Omega(g(n))$ ako postoji prirodan broj n_o i konstanta $c > 0$ tako da za sve prirodne brojeve $n >= n_0$ važi $f(n) >= cg(n)$.

Primer

- Razmotrimo funkciju $f(n)=5n^2-64n+256$.
- $f(n)=\Omega(n^2)$?

$$f(n) \geq cn^2, c=1$$
$$\Rightarrow 5n^2-64n+256 \geq n^2$$
$$\Rightarrow 4n^2-64n+256 \geq 0$$
$$\Rightarrow 4(n-8)^2 \geq 0$$
$$\forall n \geq 0, (n-8) > 0 \Rightarrow n_0 = 0$$



Polinomi i Ω

- Razmotrimo polinom

$$f(n) = \sum_{i=0}^{m} a_i n^i = a_m n^m + a_{m-1} n^{m-1} + \ldots + a_1 n^1 + a_0$$

- Onda je $f(n) = \Omega(n^m)$

Još oznaka – Θ i o (malo o)

- Θ
  - Razmotrimo funkciju $f(n)$ koja je ne-negativna za sve prirodne brojeve n >= 0.
  - Kažemo da je $f(n) = \Theta(g(n))$ ako i samo ako $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$
- Malo o
  - Razmotrimo funkciju $f(n)$ koja je ne-negativna za sve prirodne brojeve n >= 0.
  - Kažemo da je $f(n) = o(g(n))$ ako i samo ako $f(n) = O(g(n))$ and $f(n) \neq \Omega(g(n))$

Asimptotska analiza

- Primer: Hornerovo pravilo

```
1  public class Example
2  {
3      public static int horner (int[] a, int n, int x)
4      {
5          int result = a [n];
6          for (int i = n - 1; i >= 0; --i)
7              result = result * x + a [i];
8          return result;
9      }
10 }
```

statement	detailed model	simple model	big oh
5	$3\tau_{\text{fetch}} + \tau_{[\cdot]} + \tau_{\text{store}}$	5	$O(1)$
6a	$2\tau_{\text{fetch}} + \tau_{-} + \tau_{\text{store}}$	4	$O(1)$
6b	$(2\tau_{\text{fetch}} + \tau_{<}) \times (n+1)$	$3n+3$	$O(n)$
6c	$(2\tau_{\text{fetch}} + \tau_{-} + \tau_{\text{store}}) \times n$	$4n$	$O(n)$
7	$(5\tau_{\text{fetch}} + \tau_{[\cdot]} + \tau_{+} + \tau_{\times} + \tau_{\text{store}}) \times n$	$9n$	$O(n)$
8	$\tau_{\text{fetch}} + \tau_{\texttt{return}}$	2	$O(1)$
TOTAL	$(9\tau_{\text{fetch}} + 2\tau_{\text{store}} + \tau_{<} + \tau_{[\cdot]} + \tau_{+} + \tau_{\times} + \tau_{-}) \times n + (8\tau_{\text{fetch}} + 2\tau_{\text{store}} + \tau_{[\cdot]} + \tau_{-} + \tau_{<} + \tau_{\texttt{return}})$	$16n+14$	$O(n)$

Neka zapažanja

- Bez obzira kakve su konstante, asimptotska analiza uvek daje isti rezultat!
- Asimptotsko ograničenje nam daje fundamentalne informacije o vremenu izvršavanja programa.
- O ne zavisi od karakteristika računara i kompajlera na kojima se izvršava program!
- Ali ne znamo ništa o stvarnom vremenu izvršavanja programa!

Pravila za analizu velikog O (1)

- Sekvence
- Najgore vreme izračunavanja za sekvecu:

  $S_1;$

  $S_2;$

  $\vdots$

  $Sm;$

- je $O(\max(T_1(n), T_2(n), \ldots, T_m(n))$
- gde je vreme izračunavanja naredbe S_i dato sa $O(T_i(n))$

Pravila za analizu velikog O (2)

- Iteracije
- Najgore vreme izračunavanja za `for` petlju:

```
for (S1; S2; S3)
  S4;
```

- je $O(\max(T_1(n), T_2(n) \times I(n) + 1, T_3(n) \times I(n), T_4(n) \times I(n))$
- gde je vreme izračunavanja naredbe S_i dato sa $O(T_i(n))$ *a* $I(n)$ je broj iteracija za najgori slučaj

Pravila za analizu velikog O (3)

- Uslovne naredbe
- Najgore vreme izračunavanja za `if-then-else`:

```
if (S1)
  S2;
else
  S3;
```

- je $O(\max(T_1(n), T_2(n), T_3(n))$
- gde je vreme izračunavanja naredbe S_i dato sa $O(T_i(n))$

Primer: suma prvih *j* elemenata niza

```
1  public class Example
2  {
3      public static void prefixSums (int[] a, int n)
4      {
5          for (int j = n - 1; j >= 0; --j)
6          {
7              int sum = 0;
8              for (int i = 0; i <= j; ++i)
9                  sum += a[i];
10             a [j] = sum;
11         }
12     }
13 }
```

$$\sum_{i=0}^{j} a_i, 0 \leq j \leq n$$

statement	time
5a	$O(1)$
5b	$O(1) \times O(n)$ iterations
5c	$O(1) \times O(n)$ iterations
7	$O(1) \times O(n)$ iterations
8a	$O(1) \times O(n)$ iterations
8b	$O(1) \times O(n^2)$ iterations
8c	$O(1) \times O(n^2)$ iterations
9	$O(1) \times O(n^2)$ iterations
10	$O(1) \times O(n)$ iterations
TOTAL	$O(n^2)$

Realna vremena

- Vremena izvršavanja na procesoru od 1GHz, $c=1$, $n_0=0$

	$n=1$	$n=8$	$n=1K$	$n=1M$
$\Omega(1)$	1ns	1ns	1ns	1ns
$\Omega(\log n)$	1ns	3ns	10ns	20ns
$\Omega(n)$	1ns	8ns	0.102μs	1.05ms
$\Omega(n\log n)$	1ns	24ns	1.02μs	21ms
$\Omega(n^2)$	1ns	64ns	10.2μs	0.305 hours
$\Omega(n^3)$	1ns	0.512μs	1.07s	36.5 years
$\Omega(2^n)$	1ns	0.256μs	10^{292}years	10^{10000} years

Finalne napomene

- Kako verifikovati analizu algoritma?
  - Napisati program i meriti vremena za različito n
- Uporediti analizu sa realnim vremenima
- Šta ako se rezultati ne slažu sa analizom?
  - Test odnosa: recimo da smo izmerili $T(n)$ a predvideli $T(n)=O(f(n))$
  - Računamo $T(n)/f(n)$ za različite n
    - Ako odnos divergira, onda je $f(n)$ premalo
    - Ako odnos konvergira ka 0, onda je $f(n)$ preveliko
    - Ako odnos konvergira ka ne-nula konstanti, onda je $f(n)$ OK

Finalne napomene 2

- Šta ako je predviđanje preveliko?
  - Možda nismo koristili dovoljno blisko ograničenje, probamo da pronađemo odgovarajuće
  - Možda imamo korektnu analizu najgoreg vremena izvršavanja, ali se to nikad ne dešava u eksperimentu
  - Možda imamo grešku u analizi, ponoviti analizu
  - ? www.google.com? ☺