

①

Koje asimptotske notacije postoje?  
teta( $\Theta$ ), veliko O(0), malo o(0), veliko omega( $\Omega$ ), malo omega.

② Kako definišemo  $\frac{\theta}{\text{teta}}$  notaciju;  $\theta(g) = ?$

Teta služi za određivanje vremena izvršenja u najgorem slučaju; što je opisuje!

\* Uslov:  $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$  za svako  $n \geq n_0$

$\theta(g(n))$  je skup funkcija  $f(n)$

③ Kako definišemo veliko O (O-notacija)?

O-notacija se koristi za određivanje ASIMPTOTSKI GORNJE GRANICE zadate funkcije koje zavise samo od strukture programa

④ Uslov za veliko O sa gornje strane ograničava f  
 $0 \leq f(n) \leq c \cdot g(n)$  za svako  $n \geq n_0$

⑤ U matematičkom smislu u kom su odnosu veliko teta i veliko O. \* Obje su skupovi

\* Veliko teta je podskup od velikog O

⑥ Kako glasi teorema o asimptotskim notacijama.

Za bilo koje dve funkcije  $f(n)$  i  $g(n)$ , važi da je  $f(n) = \Theta(g(n))$  ako i samo ako je  $f(n) = O(g(n))$  i  $f(n) = \Omega(g(n))$ .  $\rightarrow$  veliko omega

⑦ Kako glasi rekurentna jednačina za master teoremu?

$$T(n) = aT(n/b) + f(n)$$

$a \geq 1$       } konstante       $f(n)$  - asimptotski pozitivna  
 $b > 1$       } fja  
nenegativan

⑥ Kako smo definisali rad nad grafom računaju?

- To smo definisali kao ukupno vreme računanja na 1 procesoru  $T_1$

⑦ Kako smo definisali raspon?  $T_2$

- Najveće vreme koje je potrebno da se izvrše linje duž bilo kog puta - Najduži put je kritična putanja.

⑧ Kako glasi zakon rada?

$$TP \geq \frac{T_1}{p}$$

\* Nije dovoljno meriti samo

ubrzanje, nego treba i

⑨ Kako glasi zakon raspona?

$$Tp \geq T_2$$

Bismo znali kakvi će rez.

na nekom sistemu biti za zadatim delom procesora

⑩ Kako smo definisali ubrzanje paralelnog programa?  $\frac{T_1}{Tp}$

Ubrzanje je jednako vremenu serijskog izvršenja kroz vreme na p procesora

Linearno ubrzanje  $\Theta(p)$

Savršeno ubrzanje = p

⑪ Kako glasi gornja granica za vreme  $T_p$ ?

$$\frac{T_1}{T_2} - \text{parallelizam}$$

Teorema o gornjoj granici:

ali nije to odgovor

$$TP \leq \frac{T_1}{p} + T_{\infty}$$

⑫ Kako definisemo latenciju parallelizma?

Kad parallelizam podelimo sa brojem procesora  $p$

⑬ Čemu je jednak raspon redne veze 2 podgrafa?

Jednak je zbir tih raspona pređenačnih pogradora, a raspon kod paralelne veze je maksimalan raspon od ta 2.

⑭ Čemu je jednak raspon parallel\_for petlje i ide od 1 do n.

$$\lg n = \log_2 n$$

↳ pretvara se u bin. stablo

U visina tog stabla je  $\lg n$

## 2 Pitaju za prethodna predavanja - PARALELNO

① Algoritam insertion sort, sortirajući sa umetanjem elemenata u koju klasu algoritama spada taj algoritam?  
To je inkrementalan algoritam jer inkrementalno gradi rešenja.

② Šta su prednosti, a šta mane ovog algoritma? kod malih nizova  
Prednost: efikasnost je dobra na malom broju jer raste sa kvadratom pa je kvadrat mali; \* ne koriste dodatnu memoriju već se radi in place

Mara: vreme izvršavanja se povećava mnogo; kad se nizovi povećavaju

③ Kako glasi invarianta tog niza? a od  $1 \text{ do } j-1$   
j trenutni član niza koga posmatramo, onda je niz  $a[1] \dots a[j-1]$  podniz tog niza i on je uvek sortiran.

④ Kako dokazujemo korektnost algoritama za koji imamo neku invariantu definisani?

- 1 Inicijalizacija - mora da važi pre prve iteracije
- 2 Održavanje - ako je istinita pre iteracije petlje, ostaje istinita posle iteracije, tj. pre sledeće iteracije
- 3 Završetak - kada se petlja završi, invarianta daje karakterističnu osobinu koja olakšava dokazivanje da je algoritam korektan

17.03.2012.

→ ⑤ Kako glasi završetak za ovaj naš algoritam? 5:00  
⑥ Koje je najgore izvršenje ovog algoritma tj. kako glasi vreme

asimptotski uska notacija ovog algoritma?

teta  $\Theta(n^2)$

$An^2 + Bn + C$

## Abstraktne klase

Bas

Smoraju da imaju <sup>V</sup> pure virtual metodu, u ona je prazna

\* U izvedenoj klasi mora da se implementira ta pure virtual klasa

\* Ne možemo da instanciramo abstraktnu klasu

o za klasu Shape to je `getArea();`

Npr

Shape \*shapes[] =

{  
    new Square(5),  
    new Triangle(8, 10),  
    new Square(7),  
    new Triangle(3, 4)

};

```
for (int i=0; i<4; i++)  
    cout << "shape" << i <<  
        ":" << shapes[i] ->area() << endl;
```

? Ako izvedena klasa ne pregači (override) tu pure virtual funkciju, onda ona automatski postaje i sama abstraktna klasa.

Iverzija paralelnog algoritma sortiranja  
sa spajanjem podnizova

vreme po skokom nivoj  
stabla; čvorovi kad se  
visina stabla =  $\lg n$  zaber  
zanemariti  
 $\lg n + 1$

\* za serijski algoritam  $T(n) = \Theta(n \lg n)$

↳ koristi pristup podeli i zavladaj

$$T_1(n) = \Theta(n \lg n) \quad T_1(n) = T(n) \rightarrow \text{uklonimo spawn i syns}$$

ΔRAD Δ

↑      ↳ vremenu sekv. izvršenja

Δ RASPOD (imamo 3 podgrafa (1-2, 3-5, linija 6))

$\Theta(1)$   $\Theta(n)$  ↳ ser. procedura  
if  $T_\alpha(\frac{n}{2})$  rad = raspon

$$T_\alpha(n) = \Theta(1) + T_\alpha(n/2) + \Theta(n) \quad \text{Merge-Sort} \quad \text{Merge}$$

$$T_\alpha(n) = T_\alpha(n/2) + \Theta(n) = \Theta(n)$$

$$\log_2 1 = 0 \quad f_n = \Theta(n)$$

III slučaj uslov regularnosti  
važi jer je lin. f/g

Paralelizam  $\frac{T_1(n)}{T_\alpha(n)} = \Theta(1/\lg n)$  nije tako veliki

1:15:00

Strategija podeli-i-zavladaj za paralelno spajanje podnizova

slika:

Binary-Search

$$T_1(n) - \Theta(n) (\lg n) = T_\alpha(n)$$

Analiza procedur P-Merge

$$T_\alpha(n) = T_\alpha(3n/4) + \Theta(\lg n)$$

\* osnovni slučaj  $\Theta(1) \leftarrow \text{raspon}$

$$T_\alpha(n) = \Theta(\lg^2 n)$$

rad P-Merge-Sort

$$T_1(n) = 2 T_1(n/2) + TPM_1(n)$$
$$= 2 T_1(n/2) + \Theta(n)$$

$$T_1(n) = \Theta(n \lg n)$$

$$\text{raspon} \rightarrow T_\alpha(n) = T_\alpha(n/2) + TPM_\alpha(n)$$

$$\text{ne može master} = T_\alpha(n/2) + \Theta(\lg^2 n)$$

$$\hookrightarrow T_\alpha(n) = \Theta(\lg^3 n) \quad \begin{matrix} \text{metod} \\ \text{smene} \end{matrix}$$

⑥ Algoritmi za sortiranje, 1 alg. → vveli spawn za rekursivne pozive) [koji je rad za to i rešuje?

i sync ita  
njih

$$T_1(n) = 2T_1\left(\frac{n}{2}\right) + \Theta(n)$$

$$T_1(n) = \Theta(n \cdot \lg n) \quad // \text{dručaj po master teoremi}$$

⑦ Koje je rešenje za raspon za 1 rešuje sortiranja?

$$T_2(n) = T_2\left(\frac{n}{2}\right) + \Theta(n)$$

$$T_2(n) = \Theta(n)$$

senjska procedura

// rešuje - parallelizacija procedure za spajanje

\* analiza binary search metode

raspon se malo teže računa; nismo ravnomerno raspoređeni radovi na rekursivnim pozivima

\* Najgori slučaj: kad se spaja  $\frac{3}{4} n$  elemenata i računanje rada složeno

- Izračunali smo donju granicu, gornju granicu



metodom zamene  
da bi došli do

$$\Theta(n)$$

- Onda smo prešli na analizu // rešenja gde smo koristili rezultate paralelitovanog merge-a kao fje fib u rekurenčjama za konačno rešenje

⑦ U koju klasu algoritama spada merge sort?

To je algoritam koji se zasniva na strategiji podeli i zavladači; a to se rekurentno realizuje.

⑧ Koja je prednost, a koja je manja ovog algoritma? Prednost - malo brži u odnosu na prethodni algoritam

Manja - ~~zauzimamo~~ zauzimamo dodatnu memoriju

⑨ Kako glasi invarijanta za ovaj algoritam? 8:00  
\* Sortirano uvek od  $a[p]$  do  $a[k-1]$

⑩ Kako dokazujemo korektnost ovog algoritma?

- ?  
8:00 1 Inicijalizacija -  $k=p \rightarrow$  prazan niz  $\rightarrow$  sortiran uvek  
2 Održavanje  $\xrightarrow{\text{Isluči } baci \text{ element iz levog niza}}$   $\xrightarrow{\text{društvo ubaci element iz desnog niza}}$   
3 Završetak

⑪ Kako glasi rekurenca za ovaj algoritam? rekurentno pozove samu sebe da uradi  
 $T(n) = \begin{cases} 2T(n/2) + \Theta(n), & n > 1 \\ \text{poziva se procedura spajanja} \\ \text{kombinuje mecte rezultate polovinu problema} & \end{cases}$

⑫ Zašto je  $f(n)$ -vreme merge procedure  $\Theta(n)$ ?

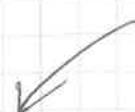
Zato što moramo da prođemo kroz ceo niz, tj. kroz  $n$  elemenata, a osnovni slučaj nosi konstantno vreme

$$\Theta(1) \cdot \underline{n \cdot \Theta(1)} = \Theta(n)$$

⑬ Koji je to slučaj u master teoremi?

- To je 2. slučaj master teoreme, a rešenje je

$$T(n) = \Theta(n \cdot \log n)$$



⑤ pa na kraju je poslednji uslov - provjeri petlje i sam izlazak  
iz petlje i onda je  $j=1$  t.j. taj podniz od 1 do  $j-1$  ceo sortiran

⑧ Šta poređimo u master metodi / poređimo  $f(n)$  sa čime?

U sva tri slučaja fja  $f(n)$  se poređi sa fjom  $n^{\log_b a}$ .

⑨ Ako su  $f$  i  $n^{\log_b a}$  asimptotski jednaki; kako glasi  $T(n)$ ?  $\hookrightarrow$  2-slučaj

$$T(n) = \Theta(n^{\log_b a} \lg n) \quad [\lg n = \log_2 n]$$

$$f(n) = \Theta(n^{\log_b a})$$

⑩ Čemu služe asimptotske notacije?

Sa njima možemo da prikažemo kako ~~vreme~~ vreme i trošek memorije zavisi od veličine problema (npr. broj elemenata niza)

- ③
- Platforma za dinamicku paralelizaciju  
 - ugnježđeni paralelizam  
 - paralelne petlje
  - Problem računanja fib. Brojeva i algoritam rekursivn.  
 Kako smo paralelizovali taj alg.  
 Koristili smo ključne reči ~~spone~~ i sync.  
 Vreme izvršenja tog sekvenčijalnog algoritma:  
 $T(n) = \Theta(F_n)$       Fibonacci  
 \* zlatni odnos
  - Kako smo definisali graf računanja?  
 Čvorovi predstavljaju neke instrukcije, tj. linije izvršavanja  
 a grane predstavljaju njihove međusobne zavisnosti.  
 Vrste grana:  
    - \* grana mreženja
    - \* grana nastavka
    - \* grana poziva
    - \* grana povratka

(15) Tačko do podataka - postoji između 2 paralelne instrukcije utoliko da jedna od njih upisuje u deluju promenljivu.
  - Kako se definije redna veza 2 linije izvršenja u ugrađenom računaru?  
 Ako u tom grafu postoji neka usmerena putanja  
 od linije  $u$  do  $v$ , onda su te linije u logičkoj  
 rednoj vezi; a ako ta putanja ne postoji onda su  
 u paralelnoj vezi.
  - Kako smo definisali sekvenčijalnu konzistentnu memo-  
 riju/deljenju?  
 \* linearno se izvršavaju tih više <sup>read</sup><sub>write</sub> operacija  
 \* odriđana se redosled u grafu računanja kao da ih je  
 1 procesor redom izvršavao.

III alg. - Strassenov metod za množenje matrica

\* umesto  $\theta(n^3)$  množenja matrica  $n/2 \times n/2$ , on obavlja  $\neq$

račun  $s_1, s_2, \dots, s_{10}$  \* u 2 ugnježdene for petlje

(rad) (vreme izvršenja za 1 korak)  $\Theta(n^2)$

račun  $C_1, C_{21}, \dots, C_{22}$  - rad  $\Theta(n^2)$

raspon - 2 ugnježden for petlje se mogu paralelizovati  $\Theta(\lg n)$

račun  $P_1, \dots, P_7$  - izvodimo kroz rekurtivne pozive  $T \cdot T\left(\frac{n}{2}\right)$

mat.  $P$

$T(n) = \begin{cases} \Theta(1), & n=1 \\ T \cdot T\left(\frac{n}{2}\right) + \Theta(n^2), & n>1 \end{cases}$  račun nad skalarima

rad koji potiče od računa za mat. i matricu  $C$ .

( $\neq$ ) rekurtivnih poziva  
svaki dejstvuje nad  
polovinom elemenata

Primenom  
master  
metode

$$T(n) = \Theta(n^{\lg 7})$$

Asimptotski brže od direktnog množenja matrice

$$\lg 7 = 3$$

$\lg 7 \rightarrow$  malo manje od 3

$$n^{\lg 7} > n^2$$

1 slučaj

\* Strassenov metod dovodi do toga da serijski algoritam je asimpt. brži od običnog alg zasnovanog na 3 for petlje.

Paralelizovan Strassenov metod - paralelizujemo for petlje za račun mat.  $S$  u  $C$ ; rekurtivne pozive paralel. - prvih 6 mogu da budu izmrešćeni - njima ćemo dodati spawn ispred d ( $\neq$ ). Če bati porvan kao obična procedura.

$s_1, s_{10}$  rad  $\Theta(n^2)$   
raspon  $\Theta(\lg n)$

Rad  $T_1(n) = \Theta(n^{\lg 7})$

Raspon  $T_0(n) = \Theta(\lg^2 n)$

Paralelizam

$$\frac{T_1(n)}{T_0(n)} = \Theta\left(\frac{n^{\lg 7}}{\lg^2 n}\right)$$

$C_1, C_{12}, \dots, C_{22}$  rad  $\Theta(n^2)$   
raspon  $\Theta(\lg n)$

Malo niti od paralelizma P-Matrix-Multiply-Recursive

Manje razgranat graf računanja (svaki čvor  $\neq$  potomaka - potrošnja memorije manja)

④ Pitana

① Koji je rad, odnosno raspon za taj direktan algoritam?

Rad i raspon 1 algoritam

↳ onoliko koliko imamo ugnježdenih petli -

$\Theta(n^2) \rightarrow$  rad da bismo izbegli tiku do početaka

$\Theta(n) \rightarrow$  raspon (prethodne 2 parallelizovane; 3. for petlja uvrštena kao serijski for)

② II algoritam - podeli i zavlači - Kako glasi rekurenca za rad za taj algoritam. Čemu je jednak  $T_1(n)$ ?

8 slučajeva / 2 ugnježdene petlje

$$T_1(n) = 8T_1\left(\frac{n}{2}\right) + \Theta(n^2) \quad \rightarrow \text{master metoda}$$

$$\text{rešenje } T_1(n) = \Theta(n^3)$$

③ Kako glasi rekurenca za raspon za 2. algoritam? <sup>raspon</sup>

$$T_{2a}(n) = T_{2a}\left(\frac{n}{2}\right) + \Theta(\lg n)$$

najveći (svi su isti za 8 parallelnih instanci)

$$\text{rešenje } T_{2a}(n) = \Theta(\lg^2 n) \quad \text{metoda zamene}$$

2 ugnježdene parallel for petlje  
 $2\Theta(\lg n)$

ii) Kako glasi rekurenca za rad za Strassenov algoritam?

$$T(n) = \begin{cases} \Theta(1), & n=1 \\ 7 \cdot T\left(\frac{n}{2}\right) + \Theta(n^2), & n>1 \end{cases}$$

master  
metoda

↳  $T(n) = \Theta(n^{\lg 7})$

⑤ Kako glasi rekurenca za raspon za Strassenov algoritam?

$$\text{rešenje: } T_{2a}(n) = \Theta(\lg^2 n)$$

\* Imamo 7 rekurzivnih poziva -  $\Theta(\frac{n}{2})$

\* Imamo 2 parallel petlje - parallel od 1 do  $n$   $\Theta(\lg n)$

~ rešavamo pomoću smene

TEST ① Šta predstavlja dekompozicija zadatka u toku pronađenja paralelizma?

② Napisati formulu za gornju granicu vremena izvršenja paralelnog algoritma na P procesora.

Napisati & posledice te teoreme.

③ Definisati labavost paralelnog algoritma koji se izvršava na idealnom paralelnom računaru sa P procesora. Koja je veza labavosti paralelizma i savremenog lin. ubrzanja? Objasni značenje elem. u jednačinama.

\* Računski raspon i paralelizam metode P-Square-Matrix-Multiply

\* Koje osobine imaju invarijanta petlje da bi se utvrdila korektnost

a)  $n^2/2 = W(n)$  DA NE

b)  $n^2/2 = W(n^2)$  DA NE

\* a)  $T(n) = 2T\left(\frac{n}{4}\right) + \sqrt{n} + 1023$

b)  $T(n) = 3T(n/4) + n \lg n$

c)  $T(n) = T(2n/3) + 1$