

LAPORAN PRAKTIKUM 3
Analisis algoritma



Sarah Navianti Dwi Sutisna
140810180021
Kelas A

PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS PADJADJARAN
2020

Latihan Analisa

1. Untuk $T(n) = 2 + 4 + 8 + 16 + \dots + n^2$, tentukan nilai C , $f(n)$, n_0 , dan notasi Big-O sedemikian sehingga $T(n) = O(f(n))$ jika $T(n) \leq C$ untuk semua $n \geq n_0$

Jawab :

1) $T(n) = 2 + 4 + 8 + 16 + \dots + 2^n$. tentukan nilai C , $f(n)$, n_0 , dan notasi Big-O sedemikian sehingga $T(n) = O(f(n))$ jika $T(n) \leq C$ untuk semua $n \geq n_0$.

Jawab :

Bentuk deret Geometri :

$$\frac{a(r^n - 1)}{r - 1} = \frac{2(2^n - 1)}{2 - 1} = 2^{n+1} - 2$$

Notasi big-O $\rightarrow O(2^n)$

$$T(n) \leq C \cdot 2^n$$

$$2^{n+1} - 2 \leq C \cdot 2^n$$

$$\frac{2^{n+1}}{2^n} - \frac{2}{2^n} \leq C$$

$$2 - \frac{2}{2^n} \leq C, \quad n_0 = 1$$

$$C \geq 1$$

2. Buktikan bahwa untuk konstanta-konstanta positif p , q , dan r :

$T(n) = pn^2 + qn + r$ adalah $O(n^2)$, $\Omega(n^2)$, $\Theta(n^2)$

Jawab :

Jawab :

• Pembuktian Big-O ($O(n^2)$)

$$T(n) \leq C \cdot F(n)$$

$$pn^2 + qn + r \leq C \cdot n^2$$

$$\frac{pn^2}{n^2} + \frac{qn}{n^2} + \frac{r}{n^2} \leq C, \quad n_0 = 1$$

$$p + q + r \leq C, \quad \text{untuk } p=1, q=1, r=1$$

$$C \geq 3$$

• Pembuktian big- Ω ($\Omega(n^2)$)

$$T(n) \geq C \cdot F(n)$$

$$pn^2 + qn + r \geq C \cdot n^2$$

$$\frac{pn^2}{n^2} + \frac{qn}{n^2} + \frac{r}{n^2} \geq C, \quad \text{untuk } n_0 = 1$$

$$C \leq p + q + r, \quad \text{untuk } p=1, q=1, r=1$$

$$C \leq 3$$

• Pembuktian big- Θ

↳ karena big-O dan big- Ω benar dan berderajat yang sama juga yaitu $O(n^2)$ dan $\Omega(n^2)$ maka big- Θ juga sama yaitu $\Theta(n^2)$ terbukti benar.

3. Tentukan waktu kompleksitas asimptotik (Big-O, Big-Ω, dan Big-Θ) dari kode program berikut:

```
for k ← 1 to n do
  for i ← 1 to n do
    for j ← 1 to n do
       $w_{ij} \leftarrow w_{ij} \text{ or } w_{ik} \text{ and } w_{kj}$ 
    endfor
  endfor
endfor
```

Jawab :

Jawab :
Kompleksitas Waktu
Operasi Assignment
 $w_{ij} \leftarrow w_{ij} \text{ or } w_{ik} \text{ and } w_{kj} = n^3$
Maka : $T(n) = n^3$

- Untuk big-O $\rightarrow O(n^3)$
 $T(n) \leq c \cdot (g(n))$
 $n^3 \leq c \cdot n^3$
 $c \geq 1$
- Untuk big-Ω $\rightarrow \Omega(n^3)$
 $T(n) \geq c \cdot (F(n))$
 $n^3 \geq c \cdot n^3$
 $c \leq 1$
- Untuk Big-Θ $\rightarrow \Theta(n^3)$
↳ Karena big-O dan big-Ω berderajat sama
yaitu $O(n^3)$ dan $\Omega(n^3)$ maka big-Θ juga
sama yaitu $\Theta(n^3)$

4. Tulislah algoritma untuk menjumlahkan dua buah matriks yang masing-masing berukuran $n \times n$. Berapa kompleksitas waktunya $T(n)$? dan berapa kompleksitas waktu asimptotiknya yang dinyatakan dalam Big-O, Big- Ω , dan Big- Θ ?

Jawab :

Jawab :

• Algoritma Penjumlahan dua matriks

```

For i ← 1 to n do
  For j ← 1 to n do
    mij ← aij + bij
  endfor
endfor

```

$T(n) = n^2$

- Untuk big O $\rightarrow O(n^2)$
 $n^2 \leq c \cdot n^2$
 $c \geq 1$
- Untuk big $\Omega \rightarrow \Omega(n^2)$
 $n^2 \geq c \cdot n^2$
 $c \leq 1$
- Untuk big $\Theta \rightarrow \Theta(n^2)$
 ↳ Karena big O dan big Ω berderajat sama yaitu $O(n^2)$ dan $\Omega(n^2)$ maka big Θ yaitu $\Theta(n^2)$

5. Tulislah algoritma untuk menyalin (copy) isi sebuah larik ke larik lain. Ukuran elemen larik adalah n elemen. Berapa kompleksitas waktunya $T(n)$? dan berapa kompleksitas waktu asimptotiknya yang dinyatakan dalam Big-O, Big- Ω , dan Big- Θ ?

Jawab :

• Algoritma menyalin larik

```

For i ← 1 to n do
  ai ← bi
endfor

```

$\rightarrow n = T(n)$

- Untuk Big O
 $n \leq cn$
 $1 \leq c$
 $c \geq 1$
- Untuk Big Ω
 $n \geq cn$
 $1 \geq c$
 $c \leq 1$
- Karena big O dan big Ω sama yaitu $O(n)$ dan $\Omega(n)$ berderajat sama maka Big Θ nya sama yaitu $\Theta(n)$

6. Diberikan algoritma Bubble Sort sebagai berikut:

```
procedure BubbleSort(input/output  $a_1, a_2, \dots, a_n$ ; integer)
{ Mengurut tabel integer TabInt[1..n] dengan metode pengurutan bubble-
sort
  Masukan:  $a_1, a_2, \dots, a_n$ 
  Keluaran:  $a_1, a_2, \dots, a_n$  (terurut menaik)
}
Deklarasi
  k : integer    { indeks untuk traversal tabel }
  pass : integer { tahapan pengurutan }
  temp : integer { peubah bantu untuk pertukaran elemen tabel }
Algoritma
  for pass  $\leftarrow$  1 to n - 1 do
    for k  $\leftarrow$  n downto pass + 1 do
      if  $a_k < a_{k-1}$  then
        { pertukarkan  $a_k$  dengan  $a_{k-1}$  }
        temp  $\leftarrow$   $a_k$ 
         $a_k \leftarrow a_{k-1}$ 
         $a_{k-1} \leftarrow$  temp
      endif
    endfor
  endfor
```

- Hitung berapa jumlah operasi perbandingan elemen-elemen tabel!
- Berapa kali maksimum pertukaran elemen-elemen tabel dilakukan?
- Hitung kompleksitas waktu asimptotik (Big-O, Big- Ω , dan Big- Θ) dari algoritma Bubble Sort tersebut!

Jawab :

6) Diberikan algoritma bubble sort

a) Hitung berapa jumlah operasi perbandingan elemen - elemen tabel!

Jawab:

Jumlah operasi perbandingan

$$1 + 2 + 3 + 4 + \dots + (n-1) = \frac{n(n-1)}{2} \text{ kali}$$

b) Berapa kali maksimum pertukaran elemen - elemen tabel dilakukan?

Jawab:

$$\frac{n(n-1)}{2} \text{ kali}$$

c) Hitung kompleksitas waktu asimtotiknya!

• Best Case (semua telah terurut)

$$\frac{(n-1)n}{2} \text{ kali, } T_{\min}(n) = \frac{n(n-1)}{2} = \frac{n^2 - n}{2}$$

• Worst Case (semua data harus diukir)

$$\text{perbandingan} \rightarrow \frac{n(n-1)}{2}$$

$$\text{memasukan nilai} \rightarrow \frac{3n(n-1)}{2}$$

$$T_{\max}(n) = \frac{4n(n-1)}{2} = 2n^2 - 2n$$

• Big O

$$2n^2 - 2n \leq cn^2$$

$$2 - \frac{2}{n} \leq c \rightarrow \text{untuk } n \geq 1$$

$$2 - 2 \leq c$$

$$c \geq 0 //$$

• Big Ω

$$\frac{n^2 - n}{2} \geq cn^2$$

$$\frac{1}{2} - \frac{1}{2n} \geq c, n_0 = 1$$

$$\frac{1}{2} - \frac{1}{2} \geq c$$

$$c \leq 0 //$$

• Big Θ → karena big O dan Big Ω berderajat sama maka big Θ sama n^2 //

7. Untuk menyelesaikan problem X dengan ukuran N tersedia 3 macam algoritma:

- Algoritma A mempunyai kompleksitas waktu $O(\log N)$
- Algoritma B mempunyai kompleksitas waktu $O(N \log N)$
- Algoritma C mempunyai kompleksitas waktu $O(N)$

Untuk problem X dengan ukuran $N=8$, algoritma manakah yang paling cepat? Secara asimptotik, algoritma manakah yang paling cepat?

7) Untuk menjelaskan problem x dgn ukuran N tersedia 3 macam algoritma :

- a) Algoritma A $\rightarrow O(\log N)$
- b) Algoritma B $\rightarrow O(N \log N)$
- c) Algoritma C $\rightarrow O(N^2)$

Jika $N=8$ maka Algoritma yang mana yang paling efektif ?

- a) $O(\log 8) = O(3 \log 2)$
- b) $O(8 \log 8) = O(24 \log 2)$
- c) $O(8^2) = O(64)$

• Maka yang paling efektif adalah Algoritma A karena semakin kecil $O()$ semakin efektif.

8. Algoritma mengevaluasi polinom yang lebih baik dapat dibuat dengan metode Horner berikut:

$$p(x) = a_0 + x(a_1 + x(a_2 + x(a_3 + \dots + x(a_{n-1} + a_n x)))) \dots)$$

function p2(input x : real) \rightarrow real
 { Mengembalikan nilai $p(x)$ dengan metode Horner }

Deklarasi

k : integer
 b_1, b_2, \dots, b_n : real

Algoritma

$b_n \leftarrow a_n$
for $k \leftarrow n - 1$ downto 0 do
 $b_k \leftarrow a_k + b_{k+1} * x$
endfor
return b_0

Hitunglah berapa operasi perkalian dan penjumlahan yang dilakukan oleh algoritma diatas, Jumlahkan kedua hitungan tersebut, lalu tentukan kompleksitas waktu asimptotik (Big-O)nya. Manakah yang terbaik, algoritma p atau p2?

Jawab :

Jawab :
Operasi memasukkan nilai
• $b_n \leftarrow a_n$ 1 kali
• $b_k \leftarrow a_k + b_k + 1 + x$ n kali
 $T(n) = n + 1$
 $O(n) = \text{untuk } p^2$
Algoritma p
Penjumlahan n kali
Perkalian n kali
 $T(n) = 2n$
↳ maka p^2 lebih baik dari p