LAPORAN PRAKTIKUM 3

ANALISIS ALGORITMA

Disusun oleh :

Muhammad Zulfikar Ali

140810180064

PROGRAM STUDI S-1 TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS PADJADJARAN

2020

$T(n) = 2+4+8+16+\dots+n^2$

$T(n) = O(f(n))$

$2+4+8+16+\dots+n^2 = O(f(n))$

$\frac{2(2^n-1)}{2-1} = O(f(n))$

①

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

notasi big $O \rightarrow O(2^n)$

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

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

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

$C \geq 1$

②

$T(n) = pn^2 + qn + r$

•> Big O

$T(n) \leq f(n)$

$pn^2+qn+r \leq C \; , \; n=1$

$p+q+r \leq C$ ⊕

•> Big Ω

$T(n) \geq f(n)$

$pn^2+qn+r \geq C(n)$, ~~[illegible]~~ $: n$

$pn+q+\frac{r}{n} \geq C.n \rightarrow n=1$

$p+q+r \geq C$ ⊕

Big θ sama dengan $O(n) = \Omega(n)$
karena memiliki orde yang sama

~~for~~ ③

```
O(n) for k ← 1 to n do
O(n)   for i ← 1 to n do
O(n)     for j ← to n do
O(1)        wij ← wij or wik and wkj
         endfor
       endfor
     endfor
```

$T(n) = O(n) + O(n) + O(n) + O(1)$

$= O(n^3) \rightarrow f(n)$

• Big-O $= O(f(n)) = O(n^3)$

• Big-Ω $= \Omega(f(n)) = \Omega(n^3)$

karena Big-O = Big-Ω, maka
Big-θ = Big-O = Big-Ω = $\theta(n^3)$

④

• Algoritma

```
for i ← 1 to n do          O(n)
  for j ← 1 to n do
    mij ← aij + bij
  endfor
endfor
```

$T(n) = O(n) + O(n)$

$= O(n^2) \rightarrow f(n)$

• Big-O

$n^2 \leq Cn^2$

$1 \leq C$

$C \geq 1$

• Big-Ω

$n^2 \geq C.n^2$

$1 \geq C$

$C \leq 1$

• Big-θ

Big-O = Big-Ω $\rightarrow$ Big-θ: $\theta(n^2$

```
5) for i ← 1 to n do        $O(n)$
        $a_i ← b_i$
   endfor
```

$T(n) = n$

$O(n)$	$\Omega(n)$
$n \le cn$	$n \ge cn$
$c \ge 1$	$c \le 1$

$O(n) = \Omega(n) \rightarrow \theta(n)$

6

a) operasi perbandingan

$$T(n) = (n-1) + (n-2) + (n-3) + \dots + 1$$

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

b) max pertukaran elemen

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

c) kompleksitas waktu

- Best case

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

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

- worst case

Perbandingan → $\frac{n(n-1)}{2}$

assignment → $\frac{3n(n-1)}{2}$

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

. Big-O

$2n^2 - 2n \le c . n^2$

$2 - \frac{2}{n} \le c$

$2 - 2 \le c$

$c \ge 0$

Big-Ω

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

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

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

$c \le 0$

$* \; O(t) = \Omega(n^2) \rightarrow \theta(n^2)$

7 a) $O(\log N)$

$O(\log 8) = O(3 \log 2)$

b) $O(N \log N)$

$O(8 \log 8)$

c) $O(N^2)$

$O(8^2) = O(64)$

Algoritma A yang paling cepat

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

- Algoritma P → jumlah n kali
  kali n kali

$T(n) = n + n = 2n$

- Algoritma P_2 → $T_2(n) = 1 + n$
  $= O(n)$

- Keduanya sama baik, karena big-O keduanya sama-sama $O(n)$