LAPORAN PRAKTIKUM 3

ANALISIS ALGORITMA

OLEH:

SHANIA SALSABILA
140810180014

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

1. Untuk $T(n) = 2 + 4 + 6 + 8 + 16 + \cdots + n^2$, tentukan nilai C, f(n), n_o, dan notasi Big-O sedemikian sehingga $T(n) = O(f(n))$ jika $T(n) \leq C\ untuk\ semua\ n\ \geq n_0$

$$T(n) = 2 + 4 + 8 + 16 + \dots + 2^n$$
$$= \frac{2(2^n - 1)}{(2-1)}$$
$$= 2(2^n - 1)$$
$$= 2^{n+1} - 2$$
$$T(n) = 2^{n+1} - 2 = O(2^n)$$
$$T(n) \leq c\ f(n)$$
$$2^{n+1} - 2 \leq c\ 2^n$$
$$2 \cdot 2^n - 2 \leq c\ 2^n$$
$$2 - \frac{2}{2^n} \leq c, \quad n_0 = 1$$
$$2 - 1 \leq c$$
$$c \geq 1$$

2. Buktikan bahwa untuk konstanta-konstanta positif p, q, dan r:
   $T(n) = pn^2 + qn + r$ adalah $O(n^2), \Omega(n^2), dan\ \Theta(n^2)$

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

→ $O(n^2)$ → Big O

$$T(n) \leq c \cdot f(n)$$
$$pn^2 + qn + r \leq c.n^2$$
$$p + \frac{q}{n} + \frac{r}{n} \leq c.n^2, \quad n_0 = 1$$
$$p + q + r \leq c$$
$$c \geq p + q + r$$

→ $\Omega(n^2)$ → big Ω

$$T(n) \geq c \cdot f(n)$$
$$pn^2 + qn + r \geq c.n$$
$$pn + q + \frac{r}{n} \geq c$$
$$p + q + r \geq c$$
$$c \leq p + q + r$$

→ Karena Big O = Big Ω = n^2
maka Big Θ = n^2

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 ← to n do
            w_ij ← w_ij or w_ik and w_kj
        endfor
    endfor
endfor
```

```
for k ← i to n do
for i ← i to n do
for j ← to n do
  wij ← wij or wik or
        wkj → n.n.n
    T(n) = n^3
  end for
  end for
  end for
```

Big O
$n^3 \le c \cdot n^3$
$1 \le c$
$c \ge 1$

Big Ω
$n^3 \ge c \cdot n^3$
$c \le 1$

Big Θ
Big O = Big Ω
maka Big $\Theta = \Theta(n^3)$

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

```
for i ← i to n do
for j ← i to n do
mij ← aij + bij  ⇒  n.n  ⇒  T(n) = n^2
 end for
 end for
```

Big O
$n^2 \le cn^2$
$1 \le c$
$c \ge 1$

Big Ω
$n^2 \ge c \cdot n^2$
$1 \ge c$
$c \le 1$

Big Θ
Big O = Big Ω
maka Big $\Theta = \Theta(n^2)$

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?

```
for i ← i to n do
  ai ← bi  ⇒ n = T(n)
end for
```

Big O
$n \le cn$
$1 \le c$
$c \ge 1$

Big Ω
$n \ge cn$
$1 \ge c$
$c \le 1$

Big Θ
Big O = Big Ω
Big $\Theta = \Theta(n)$

6. Diberikan algoritma Bubble Sort sebagai berikut:

```
procedure BubbleSort(input/output a1, a2, ..., an : integer)
{ Mengurut tabel integer TabInt[1..n] dengan metode pengurutan bubble-
sort
  Masukan: a1, a2, ..., an
  Keluaran: a1, a2, ..., an (terurut menaik)
}
Deklarasi
   k : integer     { indeks untuk traversal tabel }
   pass : integer { tahapan pengurutan }
   temp : integer { peubah bantu untuk pertukaran elemen tabel }
Algoritma
   for pass ← 1 to n - 1 do
     for k ← n downto pass + 1 do
       if ak < ak-1 then
           { pertukarkan ak dengan ak-1 }
           temp ← ak
           ak ← ak-1
           ak-1←temp
       endif
     endfor
   endfor
```

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

a) jumlah operasi perbandingan

$1 + 2 + 3 + 4 + \dots + \cancel{(n+1)}\ (n-1)$

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

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

c) Best case

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

Worst case

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

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

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

Big O

$2n^2 - 2n \leq Cn^2$

$2 - \frac{2}{n} \leq C \rightarrow n0 = 1$

$2 - 2 \leq c$

$c \geq 0$

Big Ω

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

$\frac{1}{2} - \frac{1}{2}n \geq c,\ n0 = 1$

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

$c \leq 0$

7. Untuk menyelesaikan problem X dengan ukuran N tersedia 3 macam algoritma:
   (a) Algoritma A mempunyai kompleksitas waktu O(log N)
   (b) Algoritma B mempunyai kompleksitas waktu O(N log N)
   (c) Algoritma C mempunyai kompleksitas waktu O(N^2)
   Untuk problem X dengan ukuran N=8, algoritma manakah yang paling cepat? Secara asimptotik, algoritma manakah yang paling cepat?

Uji

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:

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

```
function p2(input x : real) → real
{ Mengembalikan nilai p(x) dengan metode Horner}

Deklarasi
  k : integer
  b_1, b_2, ..., b_n : real

Algoritma
  b_n ← a_n
  for k ← n - 1 downto 0 do
     b_k ← 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?

→ bn ← an (1 kali)

→ bn ← ax + bx + 1 + x (n kali)

T(n) = n + 1

O(n) = untuk p^2

algoritma p → penjumlahan n kali

perkalian n kali

T(n) : 2n

maka p^2 lebih baik dari p