

Muhammad Iqbal Alif Fadilla
140810180020
Tugas 4

Analisis Algoritma

Tugas 4



Dibuat oleh:
Muhammad Iqbal Alif Fadilla
140810180020

**Universitas Padjadjaran
Fakultas Matematika dan Ilmu Pengetahuan
2020**

Worksheet 4

Studi Kasus

Studi Kasus 1: MERGE SORT

Setelah Anda mengetahui Algoritma Merge-Sort mengadopsi paradigma divide & conquer, lakukan Hal berikut:

1. Buat program Merge-Sort dengan bahasa C++
2. Kompleksitas waktu algoritma merge sort adalah $O(n \lg n)$. Cari tahu kecepatan komputer Anda dalam memproses program. Hitung berapa running time yang dibutuhkan apabila input untuk merge sort-nya adalah 20?

- Program C++

```
1.  /*
2.      Nama      : Muhammad Iqbal Alif Fadilla
3.      Kelas     : B
4.      NPM      : 140810180020
5.      Deskripsi : Merge Sort
6.  */
7.
8.  #include <iostream>
9.  #include <chrono>
10. using namespace std;
11.
12. void satu(int *in, int p, int q, int r)
13. {
14.     int n1 = q - p + 1;
15.     int n2 = r - q;
16.     int L[n1 + 1];
17.     int R[n2 + 1];
18.     for (int i = 1; i <= n1; i++)
19.     {
20.         L[i - 1] = in[(p - 1) + i - 1];
21.     }
22.
23.     for (int j = 1; j <= n2; j++)
24.     {
25.         R[j - 1] = in[(q - 1) + j];
26.     }
27.
28.     int i = 0;
29.     int j = 0;
30.     L[n1] = 2147483647;
31.     R[n2] = 2147483647;
32.
33.     for (int k = (p - 1); k < r; k++)
34.     {
35.         if (L[i] <= R[j])
36.         {
37.             in[k] = L[i];
38.             i = i + 1;
39.         }
40.         else
41.         {
42.             in[k] = R[j];
43.             j = j + 1;
44.         }
45.     }
```

```
46. }
47.
48. void msort(int *in, int p, int r)
49. {
50.     int q;
51.     if (p < r)
52.     {
53.         q = (p + r) / 2;
54.         msort(in, p, q);
55.         msort(in, q + 1, r);
56.
57.         satu(in, p, q, r);
58.     }
59. }
60.
61. void input(int *a, int &n)
62. {
63.     cout << "Input banyak data: ";
64.     cin >> n;
65.     cout << "\n";
66.     for (int i = 0; i < n; i++)
67.     {
68.         cout << "Input angka ke-" << i + 1 << " : ";
69.         cin >> a[i];
70.     }
71. }
72.
73. int main()
74. {
75.     int in[100];
76.     int n;
77.     input(in, n);
78.     auto start = chrono::steady_clock::now();
79.     msort(in, 1, n);
80.     auto end = chrono::steady_clock::now();
81.     cout << "\nHasil: ";
82.     for (int i = 0; i < n; i++)
83.     {
84.         cout << in[i] << " ";
85.     }
86.
87.     auto elapsed = chrono::duration_cast<chrono::nanoseconds>(end - start);
88.
89.     cout << endl;
90.     cout << "\nElapsed time in nanoseconds : " << elapsed.count() << " ns" << endl;
91.
92.     return 0;
93. }
```

```
d:\Kuliah\Semester 4\Analisis Algoritma\AnalgoKu\AnalgoKu4>merge
Input banyak data: 20

Input angka ke-1 : 2
Input angka ke-2 : 6
Input angka ke-3 : 1
Input angka ke-4 : 3
Input angka ke-5 : 20
Input angka ke-6 : 19
Input angka ke-7 : 15
Input angka ke-8 : 7
Input angka ke-9 : 18
Input angka ke-10 : 11
Input angka ke-11 : 14
Input angka ke-12 : 17
Input angka ke-13 : 9
Input angka ke-14 : 12
Input angka ke-15 : 10
Input angka ke-16 : 4
Input angka ke-17 : 8
Input angka ke-18 : 5
Input angka ke-19 : 13
Input angka ke-20 : 16

Hasil: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Elapsed time in nanoseconds : 0 ns
```

- Running Time
Sudah dicoba menggunakan steady clock, high resolution, system clock. Tetap 0 ns.
Mungkin error atau memang terlalu cepat

Studi Kasus 2: SELECTION SORT

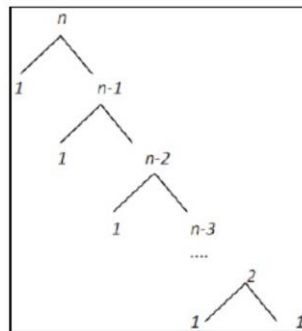
Selection sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma selection sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma selection sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) selection sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode recursion-tree** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma selection sort dengan menggunakan bahasa C++

$$T(n) = \{\theta(1) T(n-1) + \theta(n)\}$$



$$\begin{aligned} T(n) &= cn + cn-c + cn-2c + \dots + 2c + cn \\ &= c((n-1)(n-2)/2) + cn \\ &= c((n^2-3n+2)/2) + cn \\ &= c(n^2/2) - (3n/2) + 1 + cn \\ &= O(n^2) \end{aligned}$$

$$\begin{aligned} T(n) &= cn + cn-c + cn-2c + \dots + 2c + cn \\ &= c((n-1)(n-2)/2) + cn \\ &= c((n^2-3n+2)/2) + cn \\ &= c(n^2/2) - (3n/2) + 1 + cn \\ &= \Omega(n^2) \end{aligned}$$

$$\begin{aligned} T(n) &= cn^2 \\ &= \Theta(n^2) \end{aligned}$$

- Program C++

```
• /*  
•     Nama      : Muhammad Iqbal Alif Fadilla  
•     Kelas     : B  
•     NPM       : 140810180020  
•     Deskripsi  : Selection Sort  
• */  
•
```

```
• #include <iostream>
• using namespace std;
•
• int arr[100], arr2[100];
• int n;
•
• void swap(int a, int b)
• {
•     int t;
•     t = arr[b];
•     arr[b] = arr[a];
•     arr[a] = t;
• }
•
• void SelectionSort()
• {
•     int pos, i, j;
•     for (i = 1; i <= n - 1; i++)
•     {
•         pos = i;
•         for (j = i + 1; j <= n; j++)
•         {
•             if (arr[j] < arr[pos])
•                 pos = j;
•         }
•         if (pos != i)
•             swap(pos, i);
•     }
• }
•
• int main()
• {
•     cout << "Masukkan banyak data : ";
•     cin >> n;
•     cout << "\n";
•
•     for (int i = 1; i <= n; i++)
•     {
•         cout << "Masukkan data ke-" << i << " : ";
•         cin >> arr[i];
•         arr2[i] = arr[i];
•     }
•
•     SelectionSort();
•     cout << "\nHasil : " << endl;
•     for (int i = 1; i <= n; i++)
•     {
•         cout << " " << arr[i];
•     }
•     return 0;
• }
```

```
d:\Kuliah\Semester 4\Analisis Algoritma\AnalgoKu\AnalgoKu4>selection
```

```
Input the amount of data : 20
```

```
Masukkan data ke-1 : 2
```

```
Masukkan data ke-2 : 6
```

```
Masukkan data ke-3 : 1
```

```
Masukkan data ke-4 : 3
```

```
Masukkan data ke-5 : 20
```

```
Masukkan data ke-6 : 19
```

```
Masukkan data ke-5 : 20
```

```
Masukkan data ke-6 : 19
```

```
Masukkan data ke-7 : 15
```

```
Masukkan data ke-8 : 7
```

```
Masukkan data ke-9 : 18
```

```
Masukkan data ke-10 : 11
```

```
Masukkan data ke-11 : 14
```

```
Masukkan data ke-12 : 17
```

```
Masukkan data ke-13 : 9
```

```
Masukkan data ke-14 : 12
```

```
Masukkan data ke-15 : 10
```

```
Masukkan data ke-16 : 4
```

```
Masukkan data ke-17 : 8
```

```
Masukkan data ke-18 : 5
```

```
Masukkan data ke-19 : 13
```

```
Masukkan data ke-20 : 16
```

```
Hasil :
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

Studi Kasus 3: INSERTION SORT

Insertion sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma insertion sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma insertion sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode substitusi** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma insertion sort dengan menggunakan bahasa C++

$$T(n) = \{\theta(1) T(n-1) + \theta(n)\}$$

$$\begin{aligned} T(n) &= cn + cn - c + cn - 2c + \dots + 2c + cn \leq 2cn^2 + cn^2 \\ &= c((n-1)(n-2)/2) + cn \leq 2cn^2 + cn^2 \\ &= c((n^2 - 3n + 2)/2) + cn \leq 2cn^2 + cn^2 \\ &= c(n^2/2) - c(3n/2) + c + cn \leq 2cn^2 + cn^2 \\ &= O(n^2) \end{aligned}$$

$$\begin{aligned} T(n) &= cn \leq cn \\ &= \Omega(n) \end{aligned}$$

$$\begin{aligned} T(n) &= (cn + cn^2)/n \\ &= \Theta(n) \end{aligned}$$

- Program C++

```
/*
    Nama      : Muhammad Iqbal Alif Fadilla
    Kelas     : B
    NPM       : 140810180020
    Deskripsi  : Insertion search
*/

#include <iostream>
using namespace std;

int arr[100], arr2[100], n;

void InsertionSort()
{
    int temp, i, j;
    for (i = 1; i <= n; i++)
    {
        temp = arr[i];
        j = i - 1;
        while (arr[j] > temp && j >= 0)
```



```
•         {  
•             arr[j + 1] = arr[j];  
•             j--;  
•         }  
•         arr[j + 1] = temp;  
•     }  
• }  
•  
• int main()  
• {  
•     cout << "Masukkan banyak data : ";  
•     cin >> n;  
•     cout << endl;  
•  
•     for (int i = 1; i <= n; i++)  
•     {  
•         cout << "Masukkan data ke-" << i << " : ";  
•         cin >> arr[i];  
•         arr2[i] = arr[i];  
•     }  
•  
•     InsertionSort();  
•     cout << "\nHasil : " << endl;  
•  
•     for (int i = 1; i <= n; i++)  
•     {  
•         cout << arr[i] << " ";  
•     }  
•     return 0;  
• }
```

```
d:\Kuliah\Semester 4\Analisis Algoritma\AnalgoKu\AnalgoKu4>selection
```

```
Input the amount of data : 20
```

```
Masukkan data ke-1 : 2
```

```
Masukkan data ke-2 : 6
```

```
Masukkan data ke-3 : 1
```

```
Masukkan data ke-4 : 3
```

```
Masukkan data ke-5 : 20
```

```
Masukkan data ke-6 : 19
```

```
Masukkan data ke-5 : 20
```

```
Masukkan data ke-6 : 19
```

```
Masukkan data ke-7 : 15
```

```
Masukkan data ke-5 : 20
```

```
Masukkan data ke-6 : 19
```

```
Masukkan data ke-7 : 15
```

```
Masukkan data ke-8 : 7
```

```
Masukkan data ke-9 : 18
```

```
Masukkan data ke-10 : 11
```

```
Masukkan data ke-11 : 14
```

```
Masukkan data ke-12 : 17
```

```
Masukkan data ke-13 : 9
```

```
Masukkan data ke-14 : 12
```

```
Masukkan data ke-15 : 10
```

```
Masukkan data ke-16 : 4
```

```
Masukkan data ke-17 : 8
```

```
Masukkan data ke-18 : 5
```

```
Masukkan data ke-19 : 13
```

```
Masukkan data ke-20 : 16
```

```
Hasil :
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

Studi Kasus 4: BUBBLE SORT

Bubble sort merupakan salah satu algoritma sorting yang berparadigma divide & conquer. Untuk membedah algoritma bubble sort, lakukan langkah-langkah berikut:

- Pelajari cara kerja algoritma bubble sort
- Tentukan $T(n)$ dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

$$T(n) = \begin{cases} \theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

- Selesaikan persamaan rekurensi $T(n)$ dengan **metode master** untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big-O, Big-Ω, dan Big-Θ
- Lakukan implementasi koding program untuk algoritma bubble sort dengan menggunakan bahasa C++

$$\begin{aligned} T(n) &= \{\theta(1) T(n-1) + \theta(n)\} \\ T(n) &= cn + cn - c + cn - 2c + \dots + 2c + c \leq 2cn^2 + cn^2 \\ &= c((n-1)(n-2)/2) + c \leq 2cn^2 + cn^2 \\ &= c((n^2 - 3n + 2)/2) + c \leq 2cn^2 + cn^2 \\ &= c(n^2/2) - c(3n/2) + 2c \leq 2cn^2 + cn^2 \\ &= O(n^2) \end{aligned}$$

$$\begin{aligned} T(n) &= cn + cn - c + cn - 2c + \dots + 2c + c \leq 2cn^2 + cn^2 \\ &= c((n-1)(n-2)/2) + c \leq 2cn^2 + cn^2 \\ &= c((n^2 - 3n + 2)/2) + c \leq 2cn^2 + cn^2 \\ &= c(n^2/2) - c(3n/2) + 2c \leq 2cn^2 + cn^2 \\ &= \Omega(n^2) \end{aligned}$$

$$\begin{aligned} T(n) &= cn^2 + cn^2 \\ &= \Theta(n^2) \end{aligned}$$

- Program C++

```
/*
 * Nama      : Muhammad Iqbal Alif Fadilla
 * Kelas     : B
 * NPM      : 140810180020
 * Deskripsi : Bubble Sort
 */
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
    int arr[100], n, temp;
    cout << "Masukkan banyak data : ";
    cin >> n;
    cout << "\n";
```

```
•   for (int i = 0; i < n; ++i)
•   {
•       cout << "Masukkan data ke-" << i + 1 << " : ";
•       cin >> arr[i];
•   }
•
•   for (int i = 1; i < n; i++)
•   {
•       for (int j = 0; j < (n - 1); j++)
•       {
•           if (arr[j] > arr[j + 1])
•           {
•               temp = arr[j];
•               arr[j] = arr[j + 1];
•               arr[j + 1] = temp;
•           }
•       }
•   }
•   cout << "\nHasil : ";
•   for (int i = 0; i < n; i++)
•   {
•       cout << " " << arr[i];
•   }
• }
```

```
d:\Kuliah\Semester 4\Analisis Algoritma\AnalgoKu\AnalgoKu4>bubble
Masukkan banyak data : 20

Masukkan data ke-1 : 2
Masukkan data ke-2 : 6
Masukkan data ke-3 : 1
Masukkan data ke-4 : 3
Masukkan data ke-5 : 20
Masukkan data ke-6 : 19
Masukkan data ke-7 : 15
Masukkan data ke-8 : 7
Masukkan data ke-9 : 18
Masukkan data ke-10 : 11
Masukkan data ke-11 : 14
Masukkan data ke-12 : 17
Masukkan data ke-13 : 9
Masukkan data ke-14 : 12
Masukkan data ke-15 : 10
Masukkan data ke-16 : 4
Masukkan data ke-17 : 8
Masukkan data ke-18 : 5
Masukkan data ke-19 : 13
Masukkan data ke-20 : 16

Hasil : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```