

Lembar Kerja 4
Praktikum Analisis Algoritma



Okka Riswana 140810180032

Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Padjadjaran
2020

1 Merge Sort

1. Buat program Merge-Sort dengan bahasa C++.

Jawaban:

Implementasi dapat dilihat di [sini](#). Berikut adalah sedikit kutipannya,

```

_____ Merge Sort _____
1  template <typename RandomAccessIterator1,
2      typename RandomAccessIterator2,
3      typename Compare>
4  void merge_sort(RandomAccessIterator1 first1,
5      RandomAccessIterator1 last1,
6      RandomAccessIterator2 first2,
7      RandomAccessIterator2 last2,
8      Compare comp) {
9      const auto distance = std::distance(first1, last1);
10     if (distance < 2) {
11         return;
12     }
13     const auto middle1 = std::next(first1, distance >> 1);
14     const auto middle2 = std::next(first2, distance >> 1);
15     merge_sort(first2, middle2, first1, middle1, comp);
16     merge_sort(middle2, last2, middle1, last1, comp);
17     Sorting::merge(first2, middle2, middle2, last2, first1, comp);
18 }
```

2. Kompleksitas waktu algoritma merge sort adalah $O(n \cdot \log(n))$. Cari tahu kecepatan komputer anda dalam memproses program. Hitung berapa running time yang dibutuhkan apabila input untuk merge sort-nya adalah 20?

Jawaban:

Dari output program , didapat informasi sebagai berikut:

Output

```
1 Data Size: 20
2 Result: Sorted
3 Elapsed: 3367 nanoseconds
```

2 Selection Sort

1. Tentukan $T(n)$ dari rekurensi (pengulangan) selection sort berdasarkan penentuan rekurensi divide & conquer:

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

Jawaban:

Diberikan,

- **Divide:** untuk membagi data membutuhkan $\Theta(1)$.
- **Conquer:** Setiap rekursi membutuhkan biaya $O(n)$ untuk mencari elemen yang dipilih untuk ditukar. $T(n-1)$ dibutuhkan untuk menyelesaikan 1 *sub-problem* dengan ukuran $n-1$.
- **Combine:** karena data hanya dibagi 1 maka tidak ada proses *combine*, $\Theta(1)$.

didapat persamaan berikut,

$$T(n) = \begin{cases} \Theta(1), & \text{if } n = 1 \\ T(n-1) + \Theta(n) & \text{if } n > 1 \end{cases}$$

2. Selesaikan persamaan rekurensi $T(n)$ dengan metode recursion-tree untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big- O , Big- Ω , dan Big- Θ .

Jawaban:

Berikut adalah *recurrence tree* dari *selection sort*,

	h	Biaya
$c(n)$	0	n
↓		
$c(n-1)$	1	$n-1$
↓		
$c(n-2)$	2	$n-2$
↓		
$c(n-3)$	3	$n-3$
↓		
\vdots	\vdots	\vdots
↓		
$T(1)$	i	1

Gambar 1: *Recurrence Tree Selection Sort*

Kompleksitas waktu nya adalah:

$$\begin{aligned}
 T(n) &= c(n) + c(n-1) + c(n-2) + c(n-3) + \dots + c(1) \\
 &= \sum_{i=0}^n i \\
 &= \frac{n(n+1)}{2} \\
 &= O(n^2)
 \end{aligned}$$

3. Lakukan implementasi koding program untuk algoritma selection sort dengan menggunakan bahasa C++.

Jawaban:

Implementasi dari *Recursive Selection Sort* dapat dilihat di [sini](#). Berikut kutipannya,

```
_____ Recursive Selection Sort _____
1  template <typename RandomAccessIterator, typename Compare>
2  void recursive_selection_sort(RandomAccessIterator first,
3                                RandomAccessIterator last,
4                                Compare comp) {
5      if (first != last) {
6          auto temp = find_selection(first, last - 1, comp);
7          if (first != temp) {
8              std::iter_swap(first, temp);
9          }
10         recursive_selection_sort(first + 1, last, comp);
11     }
12 }
```

3 Insertion Sort

1. Tentukan $T(n)$ dari rekurensi (pengulangan) insertion sort berdasarkan penentuan rekurensi divide & conquer:

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

Jawaban:

Diberikan,

- **Divide:** untuk membagi data membutuhkan $\Theta(1)$.
- **Conquer:** Setiap rekursi membutuhkan biaya $O(n)$ untuk mencari elemen yang dipilih untuk ditukar. $T(n-1)$ dibutuhkan untuk menyelesaikan 1 *sub-problem* dengan ukuran $n-1$.
- **Combine:** karena data hanya dibagi 1 maka tidak ada proses *combine*, $\Theta(1)$.

didapat persamaan berikut,

$$T(n) = \begin{cases} \Theta(1), & \text{if } n = 1 \\ T(n-1) + \Theta(n) & \text{if } n > 1 \end{cases}$$

2. Selesaikan persamaan rekurensi $T(n)$ dengan metode substitusi untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big- O , Big- Ω , dan Big- Θ .

Jawaban:

$$\begin{aligned}T(n) &= T(n-1) + n \\&= [T(n-2) + n] + n \\&= [T(n-3) + n] + n + n \\&= [T(n-4) + n] + n + n \\&= T(n-i) + in\end{aligned}$$

Dari persamaan diatas didapat $i = n$, maka

$$\begin{aligned}T(n) &= T(1) + n \cdot n \\&= n^2 + 1 &= O(n^2)\end{aligned}$$

3. Lakukan implementasi koding program untuk algoritma insertion sort dengan menggunakan bahasa C++.

Jawaban:

Implementasi dari *Recursive Insertion Sort* dapat dilihat di [sini](#). Berikut kutipannya,

```
1  Recursive Insertion Sort
2  template <typename RandomAccessIterator, typename Compare>
3  void recursive_insertion_sort(RandomAccessIterator first,
4                                RandomAccessIterator last,
5                                Compare comp) {
6      if (std::distance(first, last) > 0) {
7          recursive_insertion_sort(first, last - 1, comp);
8          auto key = *(last - 1);
9          auto p = last - 2;
10         for (; p >= first && key < *p; --p) {
11             std::iter_swap(p, p + 1);
12         }
13         *(p + 1) = key;
14     }
```

4 Bubble Sort

1. Tentukan $T(n)$ dari rekurensi (pengulangan) bubble sort berdasarkan penentuan rekurensi divide & conquer:

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

Jawaban:

Diberikan,

- **Divide:** untuk membagi data membutuhkan $\Theta(1)$.
- **Conquer:** Setiap rekursi membutuhkan biaya $O(n)$ untuk mencari elemen yang dipilih untuk ditukar. $T(n-1)$ dibutuhkan untuk menyelesaikan 1 *sub-problem* dengan ukuran $n-1$.
- **Combine:** karena data hanya dibagi 1 maka tidak ada proses *combine*, $\Theta(1)$.

didapat persamaan berikut,

$$T(n) = \begin{cases} \Theta(1), & \text{if } n = 1 \\ T(n-1) + \Theta(n) & \text{if } n > 1 \end{cases}$$

2. Selesaikan persamaan rekurensi $T(n)$ dengan metode master untuk mendapatkan kompleksitas waktu asimptotiknya dalam Big- O , Big- Ω , dan Big- Θ .

Jawaban:

Tidak dapat dikerjakan melalui teorema master karena $b \leq 2$. Namun secara logika dapat diambil kompleksitas waktunya $O(n^2), \Omega(n^2), \Theta(n^2)$.

3. Lakukan implementasi koding program untuk algoritma insertion sort dengan menggunakan bahasa C++.

Jawaban:

Implementasi dari *Recursive Bubble Sort* dapat dilihat di [sini](#). Berikut kutipannya,

```
_____ Recursive Insertion Sort _____
1  template <typename RandomAccessIterator, typename Compare>
2  void recursive_bubble_sort(RandomAccessIterator first,
3                             RandomAccessIterator last,
4                             Compare comp) {
5      if (std::distance(first, last) > 1) {
6          for (auto left = first, right = first + 1; right != last;
7              ++left, ++right) {
8              if (comp(*right, *left)) {
9                  std::iter_swap(left, right);
10             }
11         }
12         recursive_bubble_sort(first, --last, comp);
13     }
14 }
```
