Nama : Diah Ayuning Tyas

NIM : 21091397013

Kelas : A

Algoritma Merge Sort ditemukan oleh John von Neumann di tahun 1945. Merge Sort termasuk paradigma algoritma divide and conquer. Hal ini dikarenakan algoritma ini melakukan pembagian struktur data sebelum kemudian dioperasi satu per satu. Intinya, algoritma ini menggunakan dua ide utama sebagai berikut:

1. Sebuah list yang kecil membutuhkan langkah yang lebih sedikit untuk pengurutan daripada sebuah list yang besar.
2. Untuk membentuk sebuah list terurut dari dua buah list terurut membutuhkan langkah yang lebih sedikit daripada membentuk sebuah list terurut dari dua buah list tak terurut.
   Contoh: hanya diperlukan satu kali traversal untuk masing-masing list jika keduanya sudah terurut.

- Fungsi Void

```
//implementasi program c++ Merge Sort
#include <iostream>
    using namespace std;
//fungsi void merge sort untuk pengurutan atau sorting
    void merge(int arr[], int l, int m, int r)
    {
        int i, j, k;
        int n1 = m - l + 1;
        int n2 = r - m;

        int L[n1], R[n2];

        for (i = 0; i < n1; i++)
            L[i] = arr[l + i];
        for (j = 0; j < n2; j++)
            R[j] = arr[m + 1 + j];

        i = 0;
        j = 0;
        k = l;
        while (i < n1 && j < n2)
        {
            if (L[i] <= R[j])
            {
                arr[k] = L[i];
                i++;
            }
            else
            {
                arr[k] = R[j];
                j++;
```

```
        }
        k++;
    }

    while (i < n1)
    {
        arr[k] = L[i];
        i++;
        k++;
    }

    while (j < n2)
    {
        arr[k] = R[j];
        j++;
        k++;
    }
}

void mergeSort(int arr[], int l, int r)
{
    if (l < r)
    {

        int m = l + (r - l) / 2;

        mergeSort(arr, l, m);
        mergeSort(arr, m + 1, r);

        merge(arr, l, m, r);
    }

}
//mencetak array
    void show(int A[], int size)
    {
        int i;
        for (i = 0; i < size; i++)
        {
            cout <<"["<< A[i]<<"]";
        }
    }
```

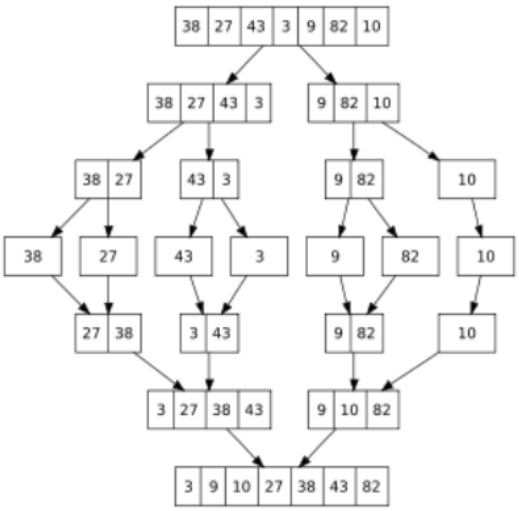
- Fungsi Main

```
//driver program
    int main()
    {
        int size;
        cout << "Masukan Banyak Data : ";
        cin >> size;
        cout << endl;

        cout << "Masukan " << size << " Nilai Data : "<<endl;
        int arr[size];

        for (int i = 0; i < size; ++i)
        {
            cin >> arr[i];
        }
        cout << endl;

        mergeSort(arr, 0, size);
    // Mencetak hasil data yang sudah di sorting
        cout << "Hasil Data yang sudah di Sorting: \n";
        show(arr, size);
        return 0;
    }
```

Algoritma Merge Sort Algoritma Merge Sort sederhananya, dapat ditulis berikut:

1. Bagi list yang tak terurut menjadi dua sama panjang atau salah satunya lebih panjang satu elemen.
2. Bagi masing-masing dari 2 sub-list secara rekursif sampai didapatkan list dengan ukuran 1.
3. Gabung 2 sublist kembali menjadi satu list terurut.

Untuk lebih jelasnya, perhatikan gambar berikut:

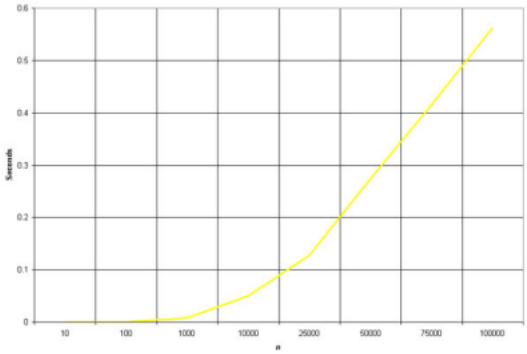


Gambar 6. Skema ini menggambarkan *Merge Sort* dalam menangani pengurutan tabel dengan 7 elemen.

Kompleksitas Merge Sort

Untuk pengurutan n elemen, merge sort memiliki kompleksitas O(nlogn) . Kompleksitas ini, juga berlaku untuk kasus terburuk. Jika running time untuk merge sort terhadap struktur data dengan banyak elemen n adalah T(n) , maka recurrence-nya T(n) = 2T(n / 2) + n berdasarkan definisi dari algoritma. Algoritma ini jelas lebih mangkus daripada algoritmaalgoritma sebelumnya. Algoritma ini rekursif, sehingga algoritma ini menjadi pilihan buruk jika ingin dijalankan di mesin komputer dengan memori yang terbatas. Perhatikan gambar! Bandingkan dengan algoritmaalgoritma dengan kompleksitas O(n2) sebelumnya yang ketika diimplementasikan umumnya membutuhkan waktu di atas 100 detik ketika menangani tabel dengan ribuan elemen.

Perhatikan gambar di bawah ini:



Gambar 7. Grafik Efisiensi *Merge Sort*

- Kelebihan Merge Sort
  - Dibanding dengan algoritma lain, merge sort ini termasuk algoritma yang sangat efisien dalam penggunaannya sebab setiap list selalu dibagi bagi menjadi list yang lebih kecil, kemudian digabungkan lagi sehingga tidak perlu melakukan banyak perbandingan.
  - Cocok untuk sorting akses datanya lambat misalnya tape drive atau hard disk.
  - Cocok untuk sorting data yang biasanya diakses secara sequentially (berurutan), misalnya linked list, tape drive, dan hard disk.

- Kekurangan Merge Sort
  - Kekurangan Merge Sort yaitu terlalu banyak menggunakan ruang pada memori.
  - Merge Sort membutuhkan lebih banyak ruang daripada jenis sorting lainnya.