K-WAY MERGE SORT

Öğrenci Adı: İclal Ertürk

Öğrenci Numarası: 21011037

Dersin Eğitmeni: M. Amaç Güvensan

Video Linki: https://www.youtube.com/watch?v=bFj3aR82scA

1- Problemin Çözümü:

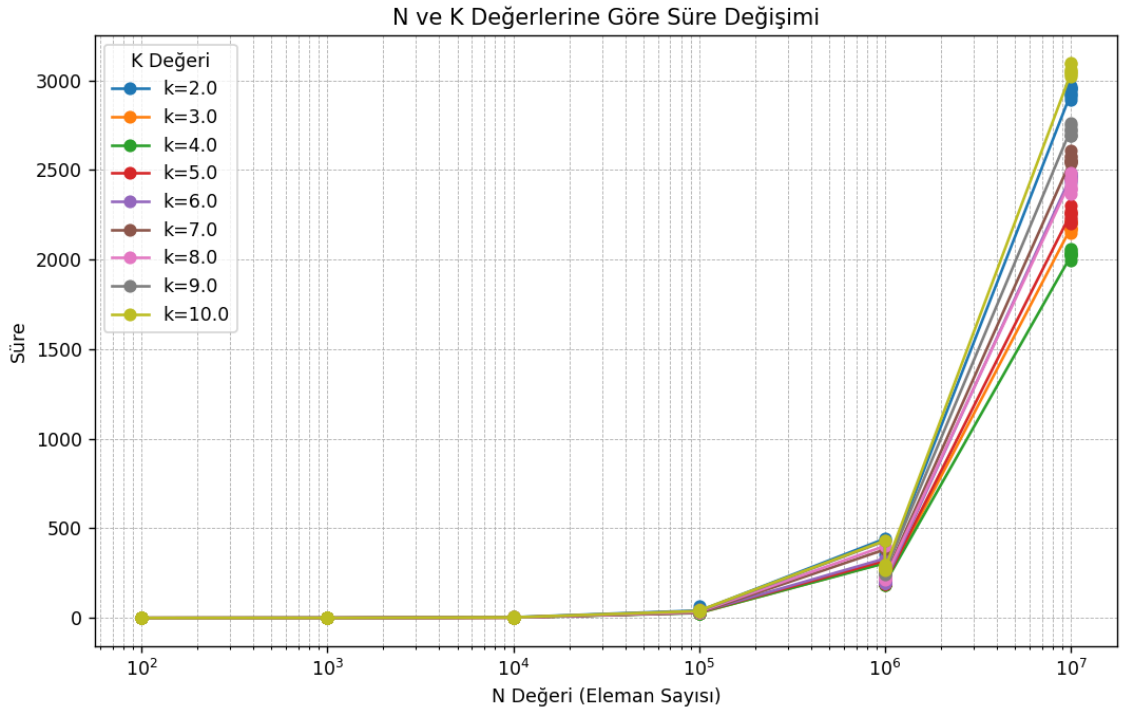
İlk olarak 6 farklı N değerinin her biri için elemanları eşsiz olan 10 farklı dizi oluşturulmuştur ve bu diziler k-way merge sort fonksiyonu ile sıralanmıştır. Her bir sıralama için ayrı ayrı süre ölçülmüştür. Ölçüm sonuçları terminal ekranına ve output.csv dosyasına yazdırılmıştır. Dosyaya yazdırılan n, k ve süre değerleri ile pythonda tablo oluşturulmuştur.

K-way merge sort algoritması her seferinde diziyi k parçaya ayırmıştır. Ayrılan parçanın boyutu kontrol edilip tekrar k parçaya ayrılmıştır. Bu işlem recursive olarak gerçekleştirilmiştir. Son dizilerin en küçük elemanları seçilerek ayrılan diziler birleştirilmiştir. Dizilerin sıralanıp sıralanmadığı yazdırılarak kontrol edilmiştir fakat bu çok uzun sürdüğü için yorum satırına alınmıştır.

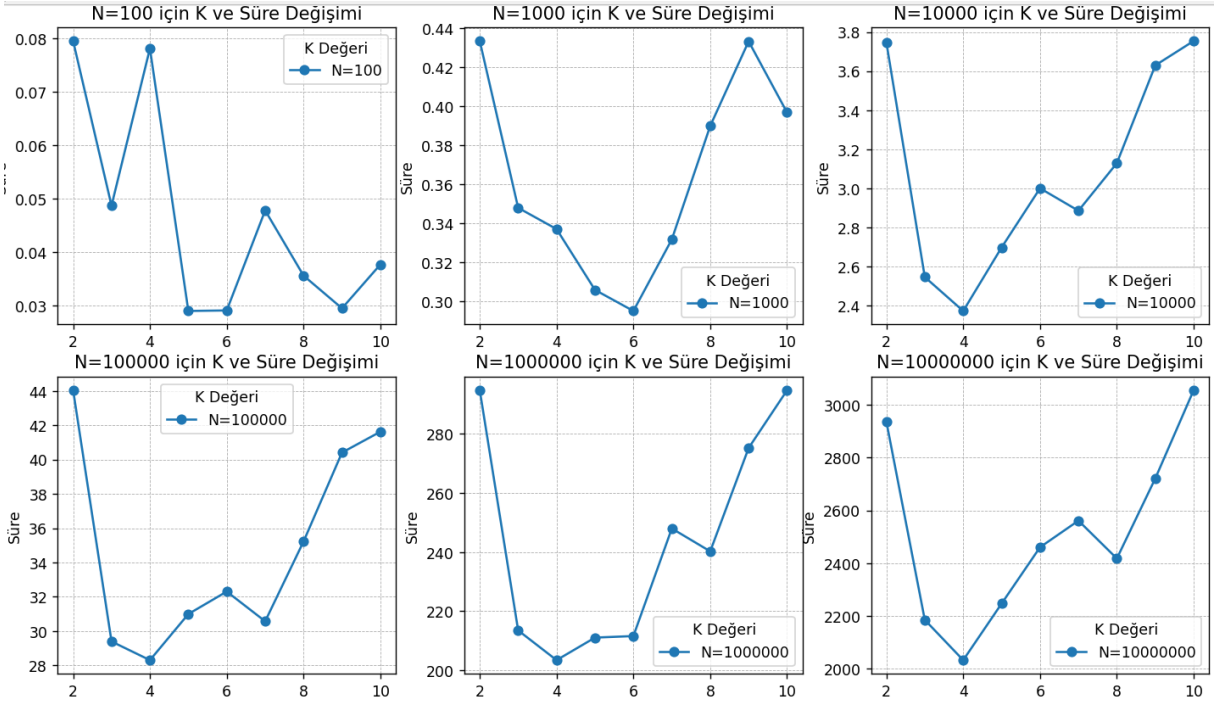
2- Karşılaşılan Sorunlar:

- ➔ Oluşturulan dizilerin elemanların random olarak üretilmesi ve 1 ve N arasında değerler alması istenmiş. Eğer diziler oluşturulurken her eklenecek eleman dizide var mı diye kontrol edip eklense idi karmaşıklık artacaktı. Bunun yerine dizileri oluştururken direkt 1'den N'e kadar olan değerler verilmiş ve dizi shuffle edilmiştir.
- ➔ K way merge sort da dizi her seferinde k parçaya bölünmekdedir. Son bölünen dizi boyutu k değerine eşit olduğunda sorunsuz çalışır fakat boyutu k değerinden küçük olduğunda oluşan dizinin sıralanma sorunu çıkmaktadır. Burada devreye normal merge sort algoritması yani 2 way merge sort girmiştir.

3- Karmaşıklık Analizi:



Dizi boyutu arttıkça çalışma süresi de artmaktadır. K'nın değeri bu artışı ufak farklılıklara sebep olmaktadır.



Farklı k değerlerinde algoritmanın çalışma süresi karşılaştırıldığında, küçük k değerleriyle sıralama daha uzun sürerken, k büyüdükçe algoritmanın çalışma süresi belirli bir k değerine kadar azalıp sonra tekrar artmaktadır. Bu durum, algoritmanın verimliliğinin k'ya duyarlı olduğunu ve k değerinin optimum bir değere sahip olabileceğini göstermektedir.

➔ Sözde kod

```
kWayMerge(array arr, integer n, integer n2, integer K)
    partSize = (n + K - 1) / K
    start = 0
    While (start < n) Do
        end = start + partSize
        If (end > n) Then set end = n
        If (end - start < K) Then
            mergeSort(arr, start, end - 1)
        Else
            kWayMerge(arr + start, end - start, n2, K)
        End If
        start = end
    End While
    result = new array[n]
```

```
    indices = new array[K]
    For i = 0 to K - 1 Do
        indices[i] = i * partSize
    End For
    index = 0
    flag = True
    While (flag) Do
        minVal = n2
        minIndex = -1
        // Alt dizilerdeki en küçük elemanı bul
        For i = 0 to K - 1 Do
            start = i * partSize
            end = start + partSize
            If (end > n) Then set end = n
            If (indices[i] < end and arr[indices[i]] < minVal) Then
                minVal = arr[indices[i]]
                minIndex = i
            End If
        End For
        // Eğer alt dizilerde eleman kalmadıysa döngüden çık
        If (minIndex == -1) Then
            flag = False
        Else
            result[index] = minVal
            indices[minIndex] += 1  // Move index of subarray with smallest element
            index += 1
        End If
    End While
    Free memory by deleting result array
End FunctionAlgoritm
```

Yukarıda sözde kodu bulunan Kway merge sort fonksiyonu algoritmadaki en karmaşık kısımdır. Karmaşıklık analizi:

Fonksiyon her çağrıda diziyi K parçaya ayırıyor ve her parçayı kWayMerge fonksiyonuyla işliyor. Bu yüzden T(n) karmaşıklığını belirlemek için aşağıdaki özyineleme denklemini yazabiliriz:

T(n)=K*T(n/K)+O(n*K)

Burada:

- K*T(n/K): K adet parçaya bölünüp her bir parça için kWayMerge fonksiyonunun çağrılması.
- O(n*K): Her seviyede K parçadan en küçük elemanları birleştirme işlemi.

Burada Master Theorem ile sonuca gidecek olursak:

a=k, b=k, d=1 => a^b=d => θ(nlogn)

Main fonksiyonu içinde tüm işlemlerin gerçekleşme kısmında da karmaşıklık artıyor.

```
For i = 0 to 6 Do
    ….
    For j = 0 to 10 Do
    …..
            For k = 2 to 10 Do
            …….
                    kwayMerge(arr[I][j],n,n,k)
            End For
    End For
End For
```

6*10*9*nlogn sonucunu elde ediyoruz ama bu rakamsal değerler her seferinde değişiyor olsa bu değerlere bağlı olarak bir sonuç çıkardı burada sabit olduklarından sonuç yine yaklaşık olarak θ(nlogn) dir.

4- Ekran Çıktıları:

```
n = 100 0.dizi k= 2 K-way merge sort suresi: 0.030 milisaniye
n = 100 0.dizi k= 3 K-way merge sort suresi: 0.015 milisaniye
n = 100 0.dizi k= 4 K-way merge sort suresi: 0.013 milisaniye
n = 100 0.dizi k= 5 K-way merge sort suresi: 0.014 milisaniye
n = 100 0.dizi k= 6 K-way merge sort suresi: 0.017 milisaniye
n = 100 0.dizi k= 7 K-way merge sort suresi: 0.016 milisaniye
n = 100 0.dizi k= 8 K-way merge sort suresi: 0.013 milisaniye
n = 100 0.dizi k= 9 K-way merge sort suresi: 0.014 milisaniye
n = 100 0.dizi k= 10 K-way merge sort suresi: 0.018 milisaniye
n = 100 1.dizi k= 2 K-way merge sort suresi: 0.020 milisaniye
n = 100 1.dizi k= 3 K-way merge sort suresi: 0.019 milisaniye
n = 100 1.dizi k= 4 K-way merge sort suresi: 0.031 milisaniye
n = 100 1.dizi k= 5 K-way merge sort suresi: 0.013 milisaniye
n = 100 1.dizi k= 6 K-way merge sort suresi: 0.030 milisaniye
n = 100 1.dizi k= 7 K-way merge sort suresi: 0.016 milisaniye
n = 100 1.dizi k= 8 K-way merge sort suresi: 0.015 milisaniye
n = 100 1.dizi k= 9 K-way merge sort suresi: 0.017 milisaniye
n = 100 1.dizi k= 10 K-way merge sort suresi: 0.016 milisaniye
n = 100 2.dizi k= 2 K-way merge sort suresi: 0.033 milisaniye
n = 100 2.dizi k= 3 K-way merge sort suresi: 0.015 milisaniye
n = 100 2.dizi k= 4 K-way merge sort suresi: 0.015 milisaniye
n = 100 2.dizi k= 5 K-way merge sort suresi: 0.013 milisaniye
n = 100 2.dizi k= 6 K-way merge sort suresi: 0.018 milisaniye
n = 100 2.dizi k= 7 K-way merge sort suresi: 0.017 milisaniye
n = 100 2.dizi k= 8 K-way merge sort suresi: 0.018 milisaniye
n = 100 2.dizi k= 9 K-way merge sort suresi: 0.015 milisaniye
n = 100 2.dizi k= 10 K-way merge sort suresi: 0.019 milisaniye
n = 100 3.dizi k= 2 K-way merge sort suresi: 0.019 milisaniye
n = 100 3.dizi k= 3 K-way merge sort suresi: 0.014 milisaniye
n = 100 3.dizi k= 4 K-way merge sort suresi: 0.017 milisaniye
n = 100 3.dizi k= 5 K-way merge sort suresi: 0.013 milisaniye
n = 100 3.dizi k= 6 K-way merge sort suresi: 0.017 milisaniye
n = 100 3.dizi k= 7 K-way merge sort suresi: 0.013 milisaniye
n = 100 3.dizi k= 8 K-way merge sort suresi: 0.016 milisaniye
n = 100 3.dizi k= 9 K-way merge sort suresi: 0.017 milisaniye
n = 100 3.dizi k= 10 K-way merge sort suresi: 0.076 milisaniye
n = 100 4.dizi k= 2 K-way merge sort suresi: 0.022 milisaniye
n = 100 4.dizi k= 3 K-way merge sort suresi: 0.015 milisaniye
```

```
n = 100 4.dizi k= 4 K-way merge sort suresi: 0.017 milisaniye
n = 100 4.dizi k= 5 K-way merge sort suresi: 0.014 milisaniye
n = 100 4.dizi k= 6 K-way merge sort suresi: 0.016 milisaniye
n = 100 4.dizi k= 7 K-way merge sort suresi: 0.016 milisaniye
n = 100 4.dizi k= 8 K-way merge sort suresi: 0.018 milisaniye
n = 100 4.dizi k= 9 K-way merge sort suresi: 0.015 milisaniye
n = 100 4.dizi k= 10 K-way merge sort suresi: 0.019 milisaniye
n = 100 5.dizi k= 2 K-way merge sort suresi: 0.020 milisaniye
n = 100 5.dizi k= 3 K-way merge sort suresi: 0.017 milisaniye
n = 100 5.dizi k= 4 K-way merge sort suresi: 0.016 milisaniye
n = 100 5.dizi k= 5 K-way merge sort suresi: 0.012 milisaniye
n = 100 5.dizi k= 6 K-way merge sort suresi: 0.015 milisaniye
n = 100 5.dizi k= 7 K-way merge sort suresi: 0.028 milisaniye
n = 100 5.dizi k= 8 K-way merge sort suresi: 0.017 milisaniye
n = 100 5.dizi k= 9 K-way merge sort suresi: 0.018 milisaniye
n = 100 5.dizi k= 10 K-way merge sort suresi: 0.018 milisaniye
n = 100 6.dizi k= 2 K-way merge sort suresi: 0.024 milisaniye
n = 100 6.dizi k= 3 K-way merge sort suresi: 0.017 milisaniye
n = 100 6.dizi k= 4 K-way merge sort suresi: 0.017 milisaniye
n = 100 6.dizi k= 5 K-way merge sort suresi: 0.044 milisaniye
n = 100 6.dizi k= 6 K-way merge sort suresi: 0.014 milisaniye
n = 100 6.dizi k= 7 K-way merge sort suresi: 0.013 milisaniye
n = 100 6.dizi k= 8 K-way merge sort suresi: 0.043 milisaniye
n = 100 6.dizi k= 9 K-way merge sort suresi: 0.016 milisaniye
n = 100 6.dizi k= 10 K-way merge sort suresi: 0.018 milisaniye
n = 100 7.dizi k= 2 K-way merge sort suresi: 0.032 milisaniye
n = 100 7.dizi k= 3 K-way merge sort suresi: 0.021 milisaniye
n = 100 7.dizi k= 4 K-way merge sort suresi: 0.018 milisaniye
n = 100 7.dizi k= 5 K-way merge sort suresi: 0.015 milisaniye
n = 100 7.dizi k= 6 K-way merge sort suresi: 0.015 milisaniye
n = 100 7.dizi k= 7 K-way merge sort suresi: 0.017 milisaniye
n = 100 7.dizi k= 8 K-way merge sort suresi: 0.017 milisaniye
n = 100 7.dizi k= 9 K-way merge sort suresi: 0.103 milisaniye
n = 100 7.dizi k= 10 K-way merge sort suresi: 0.019 milisaniye
n = 100 8.dizi k= 2 K-way merge sort suresi: 0.022 milisaniye
n = 100 8.dizi k= 3 K-way merge sort suresi: 0.019 milisaniye
n = 100 8.dizi k= 4 K-way merge sort suresi: 0.016 milisaniye
n = 100 8.dizi k= 5 K-way merge sort suresi: 0.015 milisaniye
n = 100 8.dizi k= 6 K-way merge sort suresi: 0.016 milisaniye
n = 100 8.dizi k= 7 K-way merge sort suresi: 0.015 milisaniye
```

```
n = 100 8.dizi k= 8 K-way merge sort suresi: 0.017 milisaniye
n = 100 8.dizi k= 9 K-way merge sort suresi: 0.031 milisaniye
n = 100 8.dizi k= 10 K-way merge sort suresi: 0.020 milisaniye
n = 100 9.dizi k= 2 K-way merge sort suresi: 0.020 milisaniye
n = 100 9.dizi k= 3 K-way merge sort suresi: 0.017 milisaniye
n = 100 9.dizi k= 4 K-way merge sort suresi: 0.043 milisaniye
n = 100 9.dizi k= 5 K-way merge sort suresi: 0.015 milisaniye
n = 100 9.dizi k= 6 K-way merge sort suresi: 0.015 milisaniye
n = 100 9.dizi k= 7 K-way merge sort suresi: 0.015 milisaniye
n = 100 9.dizi k= 8 K-way merge sort suresi: 0.016 milisaniye
n = 100 9.dizi k= 9 K-way merge sort suresi: 0.020 milisaniye
n = 100 9.dizi k= 10 K-way merge sort suresi: 0.018 milisaniye
n = 1000 0.dizi k= 2 K-way merge sort suresi: 0.209 milisaniye
n = 1000 0.dizi k= 3 K-way merge sort suresi: 0.143 milisaniye
n = 1000 0.dizi k= 4 K-way merge sort suresi: 0.176 milisaniye
n = 1000 0.dizi k= 5 K-way merge sort suresi: 0.149 milisaniye
n = 1000 0.dizi k= 6 K-way merge sort suresi: 0.126 milisaniye
n = 1000 0.dizi k= 7 K-way merge sort suresi: 0.146 milisaniye
n = 1000 0.dizi k= 8 K-way merge sort suresi: 0.160 milisaniye
n = 1000 0.dizi k= 9 K-way merge sort suresi: 0.169 milisaniye
n = 1000 0.dizi k= 10 K-way merge sort suresi: 0.186 milisaniye
n = 1000 1.dizi k= 2 K-way merge sort suresi: 0.198 milisaniye
n = 1000 1.dizi k= 3 K-way merge sort suresi: 0.144 milisaniye
n = 1000 1.dizi k= 4 K-way merge sort suresi: 0.140 milisaniye
n = 1000 1.dizi k= 5 K-way merge sort suresi: 0.147 milisaniye
n = 1000 1.dizi k= 6 K-way merge sort suresi: 0.125 milisaniye
n = 1000 1.dizi k= 7 K-way merge sort suresi: 0.148 milisaniye
n = 1000 1.dizi k= 8 K-way merge sort suresi: 0.163 milisaniye
n = 1000 1.dizi k= 9 K-way merge sort suresi: 0.169 milisaniye
n = 1000 1.dizi k= 10 K-way merge sort suresi: 0.182 milisaniye
n = 1000 2.dizi k= 2 K-way merge sort suresi: 0.208 milisaniye
n = 1000 2.dizi k= 3 K-way merge sort suresi: 0.148 milisaniye
n = 1000 2.dizi k= 4 K-way merge sort suresi: 0.139 milisaniye
n = 1000 2.dizi k= 5 K-way merge sort suresi: 0.156 milisaniye
n = 1000 2.dizi k= 6 K-way merge sort suresi: 0.125 milisaniye
n = 1000 2.dizi k= 7 K-way merge sort suresi: 0.152 milisaniye
n = 1000 2.dizi k= 8 K-way merge sort suresi: 0.168 milisaniye
n = 1000 2.dizi k= 9 K-way merge sort suresi: 0.174 milisaniye
n = 1000 2.dizi k= 10 K-way merge sort suresi: 0.192 milisaniye
n = 1000 3.dizi k= 2 K-way merge sort suresi: 0.546 milisaniye
```