

## Algoritma Analizi Ders 2

### EklemeLİ Sıralama

Girdi:  $n$  elementli bir sayı dizisi  $\{a_1, a_2, \dots, a_n\}$

Cıktı:  $n$  elementli verilen dizili sıralayarak

$$\{a_1, a_2, \dots, a_n\} \quad a_1 \leq a_2 \leq a_3 \leq \dots \leq a_n$$

Sıralanık istenilenin şartları da aynı zamanda şartlar (keys) olmak adlandırılabilir.

Algoritmdan: Pseudo code de formlu olarak:

Eğerde sıralanır olan element 4'tür kılınırsa bir

bırakılmalıdır.

### INSERTION SORT (A)

for  $j = 2$  to  $A.length$

key =  $A[j]$

$i = j - 1$

while  $i > 0$  and  $A[i] > key$

$A[i+1] = A[i]$

$i = i - 1$

$A[i+1] = key$

5	2	6	1	6	1	3
---	---	---	---	---	---	---



2	5	4	6	1	6	1	3
---	---	---	---	---	---	---	---



2	4	5	6	1	6	1	3
---	---	---	---	---	---	---	---

$6 > 5$  (dönüş)

2	4	5	6	1	6	1	3
---	---	---	---	---	---	---	---



1 en başa getirilir.

1	2	4	5	6	3
---	---	---	---	---	---

3 2 1 ye kadar  
key sırası

1	2	3	4	5	6
---	---	---	---	---	---

Dörgön her verkstiginda

- $A[j]$  -sun sindayoleğimiz element (ka)
  - $A[1 \dots J-1]$  e kader olsun kism sindirimistir.
  - $A[J \dots n]$  e kader olsun kism sindirim betliyac

Algoritmonun deðnu celimesim 'is potemek  
icin  $A[1, \dots, j-1]$  alt diæssinin loop  
invaniort (döngü sabiti) olarık taramak yox.

Dögs sobitten bir aforitmon dönu  
oldgum gasterat iðn en kvallobill

- Baslogeic (D&gt;V sobti d&gt;vndr ik  
valignasinde d&gt;nudr)

- Sündürme (Döge) sobitti belli bir  
döge (Gökhan) öncesi döge  
ise bir sonraki döges (Gökhan)  
öncesi depremlerini konur.)

- Sonlurma (Dünya sonlandıında dünya Sabiti blue algoritminin deprem olaylarını gösterir)

Algoritmo's maken zijn algoritmen eenvoudig te volgen en te herhalen.

Hofsta  
Norway Bedognies  
Bachwacht Zorn (Vidre  
Time (Vidre  
Zorn)

## Eğerdeki Sıralama Algoritması Analizi

- Birçoklu giriş boyutu farklılık gösterir.
  - Giriş boyutu sıralanacak eleman sayısidir.
- Birinci olasılık galatma zamanı farklılık gösterir.
  - Galatma zamanı algoritma sırasına göre değişikliklerdeki farklı operasyon veya adımlar sayısını gösterir.

Sıra Tablosu

for  $j=2$  to  $A.length$

    key =  $A[j]$

$i=j-1$

    while  $i > 0$  and  $A[i] > key$

$A[i+1] = A[i]$

$i = i - 1$

$A[i+1] = key$

$c_1$

$n$

$c_2$

$n-1$

$c_3$

$n-1$

$c_4$

$\sum_{j=2}^n t_j$

$c_5$

$\sum_{j=2}^n (t_j - 1)$

$c_6$

$\sum_{j=2}^n (t_j - 1)$

$c_7$

$n-1$

$T(n) = \text{hepsinin toplam}$

En iyi durumda, elemanlar önceki sıralamas  
halde gelebilir ve hiçbir zaman while (iudeki döngü)  
ye girmez.

$$\begin{aligned}
 T(n) &= c_1 n + c_2(n-1) + c_3(n-1) + c_4(n-1) + c_5(n-1) \\
 &= (c_1 + c_2 + c_3 + c_4 + c_5)n - (c_2 + c_3 + c_4 + c_5)
 \end{aligned}$$

$\doteq an + b$  şeklinde bininci dereceden

Bu sebeple en iyi durumda  $n^2$ 'e bağlı lineer  
bir fonksiyondur.

## En kötü durumda

Sayıları tersten sıralanmış şekilde debilsiz. Her eleman için içdeğerde düşükle maksimum defa gireriz.

$A[i, j-1]$  elemanının sıralanışı  $i < j$

$$j = 2, 3, 4, \dots, n$$

$$\sum_{j=2}^n j = \frac{n(n+1)}{2} - 1, \quad \left( \frac{n^2-n}{2} - 1 \right)$$

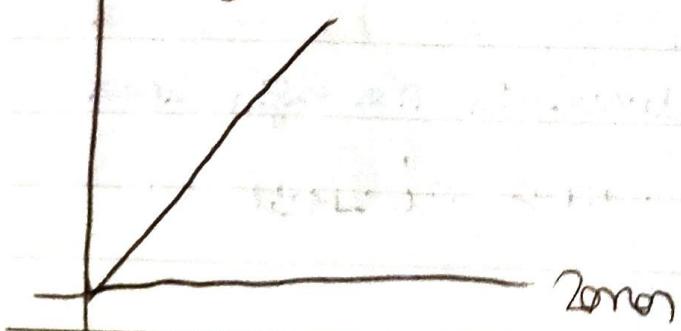
$$\sum_{j=2}^n (j-1) = \frac{n(n-1)}{2} \quad \left( \frac{n^2-n}{2} \right)$$

$$\begin{aligned} T(n) &= C_1 n + C_2(n-1) + C_3(n-1) + C_4 \left( \frac{n(n-1)}{2} - 1 \right) \\ &\quad + C_5 \left( \frac{n(n-1)}{2} \right) + C_6 \left( \frac{n(n-1)}{2} \right) + C_7(n-1) \\ &= \left( \frac{C_4}{2} + \frac{C_5}{2} + \frac{C_6}{2} \right) n^2 - \left( \frac{C_4}{2} + \frac{C_5}{2} + \frac{C_6}{2} \right) n + (C_2 + C_3 + C_4 + C_7) \\ &\quad - (C_2 + C_3 + C_4 + C_7) \end{aligned}$$

$T(n) = an^2 + bn + c$  şeklinde yazılabilir.

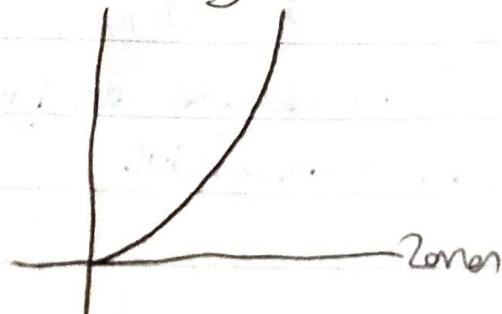
Bu sebeple belirtilen değer  $n$ 'e bağlı, quadric (ikinci dereceden) bir fonksiyondur.

Eleman sayısı



En kötü durumda

Eleman Sayısı



En iyi durumda

## Divide and Conquer (Böl ve Yarat)

Yenilik ve varaklı algoritmalernin en önemli  
recursive (yinelemeli) (kendiini çağırır fonk.)  
sözlükte, şatır.

Problem daha ufak alt problemlere bölünür  
Alt problemler çözülerek, asıl problem çözümüne  
dönur

- Böl (Asıl problemi boluşturan alt probleme  
bölüyoruz)

- Yarat (Alt problemlerin her birine recursive  
döner ismi. Alt problem yeteneği  
kadar küçük olabilir. Çoğun zaman hizaplanır)

- Birleştir (Alt problemlere belirli çözümler  
birleştirilir ve asıl probleme (hiz  
bırır)

## Birleşimeli Süreçler (Merge Sort)

- Böl:  $n$  elemanlı sıralamayı iki  $n/2$   
elemanlı böler

- Yarat: Birinci adında sıralanmış olan alt  
recursive döner aynı şekilde sıraları

- Birleştir: Birinci adında sıralanmış olan alt  
dönerin birleşiminek sıralanmış döneri  
elde et

Yarat adındaki recursive çağırılar, tek eleman kalanaya  
kadar devam eder

Ancak sadece birleştirime gerekmedi.

Dynamic Endings

P(1)

Binöstrir MERGE(A, p, q, r)

$p \leq q < r$

Elementlerin birleşen liste

Bu işlem  $A[p \dots q]$  ve  $A[q+1 \dots r]$

alt dizelerin sıralanmış eklentiler

Binöstrerek  $A[p \dots r]$  deklini oluşturur.

MERGE işlemi  $O(n)$  zamanlılıklaşırır

$$n = r - p + 1$$

Merge(A, p, q, r)

$$n_1 = q - p + 1$$

$$n_2 = r - q$$

Left[1 ...  $n_1 + 1$ ], Right[1 ...  $n_2 + 1$ ]

for  $i = 1$  to  $n_1$

$$\text{Left}[i] = A[p+i-1]$$

for  $j = 1$  to  $n_2$

$$\text{Right}[j] = A[q+j]$$

$$\text{Left}[n_1 + 1] = \infty$$

$$\text{Right}[n_2 + 1] = \infty$$

$$i = 1$$

$$j = 1$$

for  $k = p$  to  $r$

if  $\text{Left}[i] \leq \text{Right}[j]$

$$A[k] = \text{Left}[i]$$

$$i = i + 1$$

else  $A[k] = \text{Right}[j]$

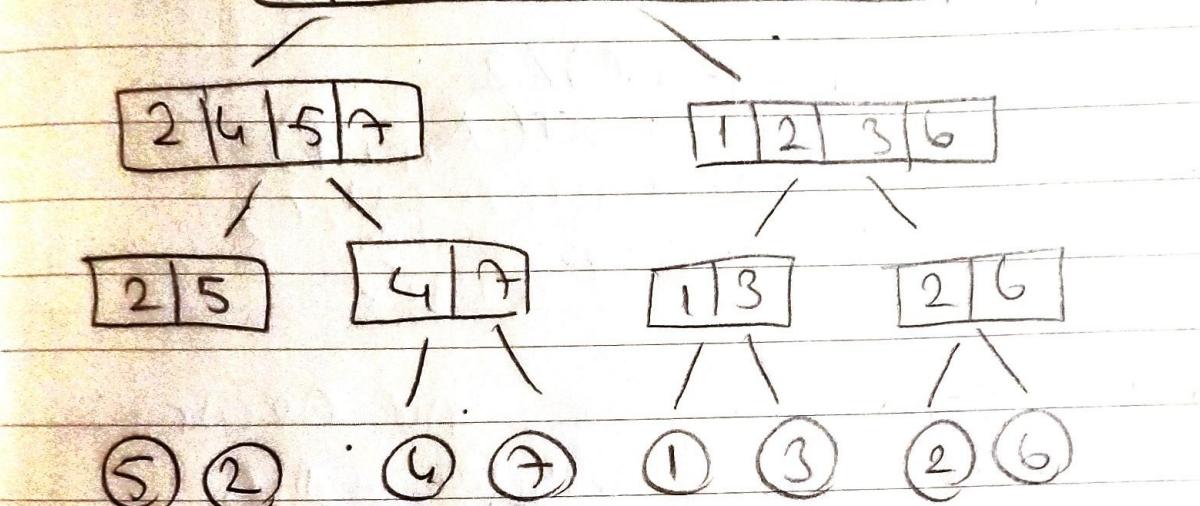
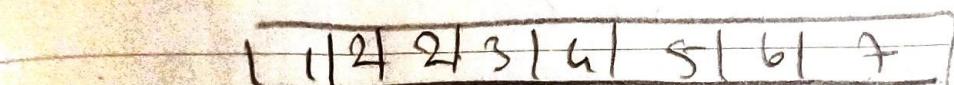
$$j = j + 1$$

MERGE fonsiyundaki döngü sabiti: (loop invariant)

Döngünün her çevresinde

$A[p \dots k-1]$  alt dizisi

$\text{Left}[1 \dots n_1+1]$  ve  $\text{Right}[1 \dots n_2+1]$  alt dizilerinin  
en son  $k-p$  elementini içermektedir



Algoritmanın döngü konusunu göstermek için

**İlk döngü:** Döngü ilk adımda  $k=p$  durumundadır. Bu durumda  $A[p \dots k-1]$  alt dizisi boştur.

Left ve Right, alt dizilerinin  $k-p=0$  tane  
en son elementini içermektedir.

$L[i]$   $R[i]$  alt dizileri: Adımların köşebelirler  
telli döndürür.

Sünderme:

$L[i] \leq R[i]$  oldugu fonksiyon A $[p \dots k-1]$   
 $k-p$  en küçük element iceriyor

$L[i]$  elementi  $A[k]$  ya yeknosturulur A $[p \dots k]$   
o��e en küçük element icerilen Tantasi içinde bulunur

Sorlarma: Döngü sorlundüğünde  $k = r + 1$   
 Döngü sonottage şöre  $A[p - k - 1]$  sadece  
 olmalıdır  $A[p - r]$  divisive döngümüşür  
 MERGE işlemi  $\Theta(n)$  zaman alır.

MERGESORT( $A, p, r$ )

if  $p < r$

$$q = (p + r) / 2$$

MERGESORT( $A, p, q$ )

MERGESORT( $A, p+1, r$ )

MERGE( $A, p, q, r$ )

- Problem boyutluğunu her recursive çağrıda iki katına çıkarır.
- Küçük problemler  $\Theta(1)$  zamanında çözülebilirler
- Küçük problem büyük problemin  $1/b$ 'si boyutluğundan
- Problem boyutu  $\Theta(n)$  zaman alır. Birlesimde  $\Theta(n)$  zaman alır.

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

## MERGE SORT ANALYSE

-**Bölüm:** Alt dizinin orta noktasını  
kesip iki yarım sublisti 2<sup>n-1</sup> tane  $D(n) = \Theta(1)$

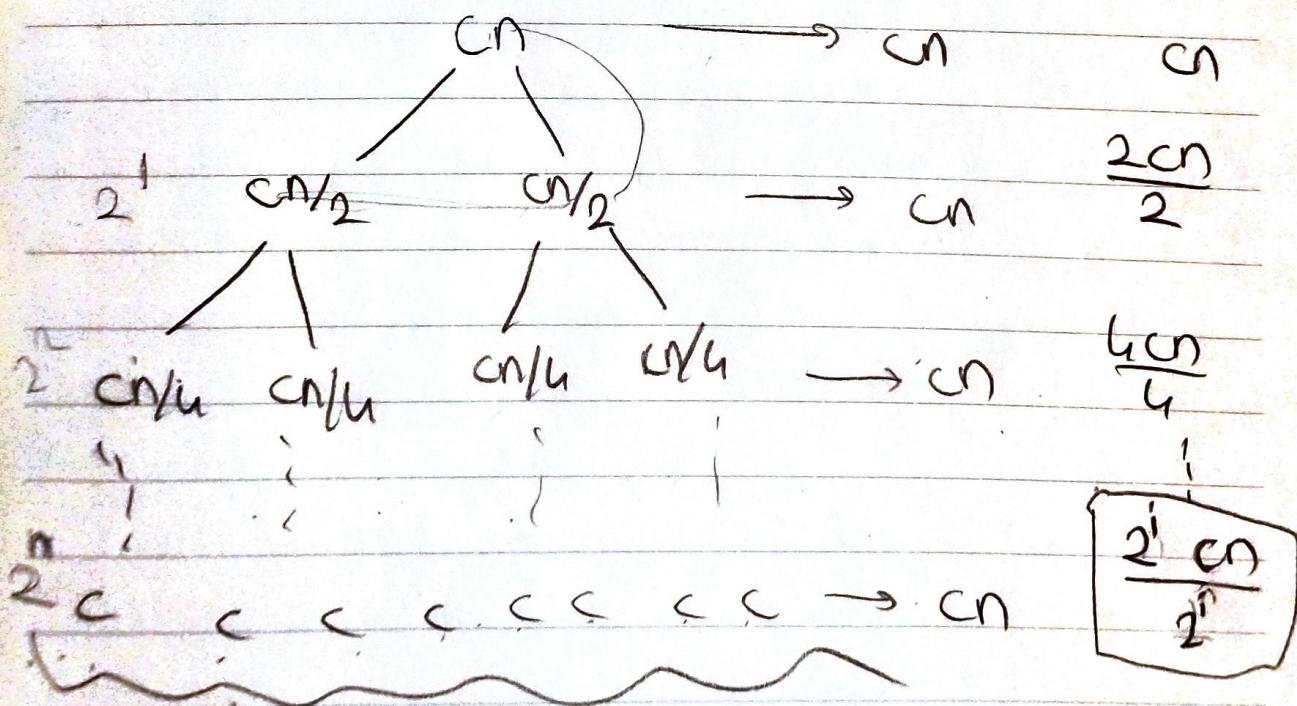
-**Yeneti:** Rekursiv şekilde düzeltilen  
iki kere problem çözülmüş problemler  
bağcık  $n/2$   $2(T(n/2))$

-**Birimleri:** Merge etmeni  $\Theta(n)$  sure  
oluş.  $C(n) = \Theta(n)$

$$T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ 2T(n/2) + \Theta(n) & \text{if } n>1 \end{cases}$$

$$T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ 2T(n/2) + cn & \text{if } n>1 \end{cases}$$

$x = b^y$  katman  
 $\log_b(x) = y$   $\Rightarrow$  sayısal



$n$  tane  
harfini  $\Theta(n)$

$$\boxed{\lg n + 1} \quad \boxed{\lg n + 1} \quad \boxed{\lg n + 1} \quad \boxed{\frac{cn(\lg n + 1)}{n \log n}}$$