4.4.6 hızlı (quick) sıralama

→ böl-yönet yaklaşımı

→ yarılama yerine başka bir yapı

kabaca algoritma

- → **pivot** değeri seç
- → liste içerisinde pivotun doğru konumunu bul: **bölme noktası**
- → bölme noktası yardımıyla listeyi ikiye **böl** (sol - sağ)
- → bu iki parçayı ayrı ayrı **özyineli** quick sort ile sırala

pivot seçimi

→ çok farklı yolları var

→ ilk elemanı seç

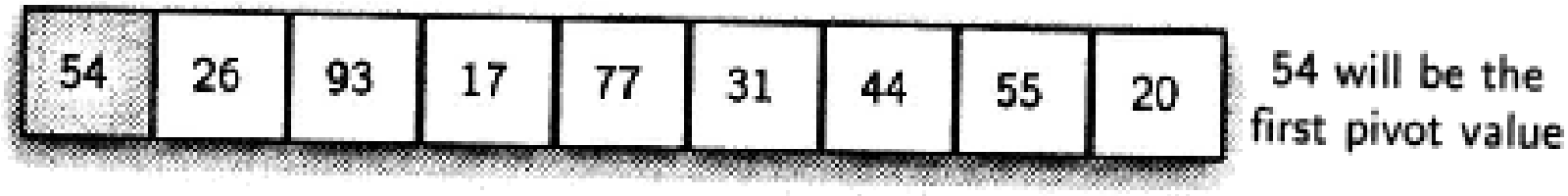


Figure 4.24: The First Pivot Value for a Quick Sort

→ rastgele bir değeri seç

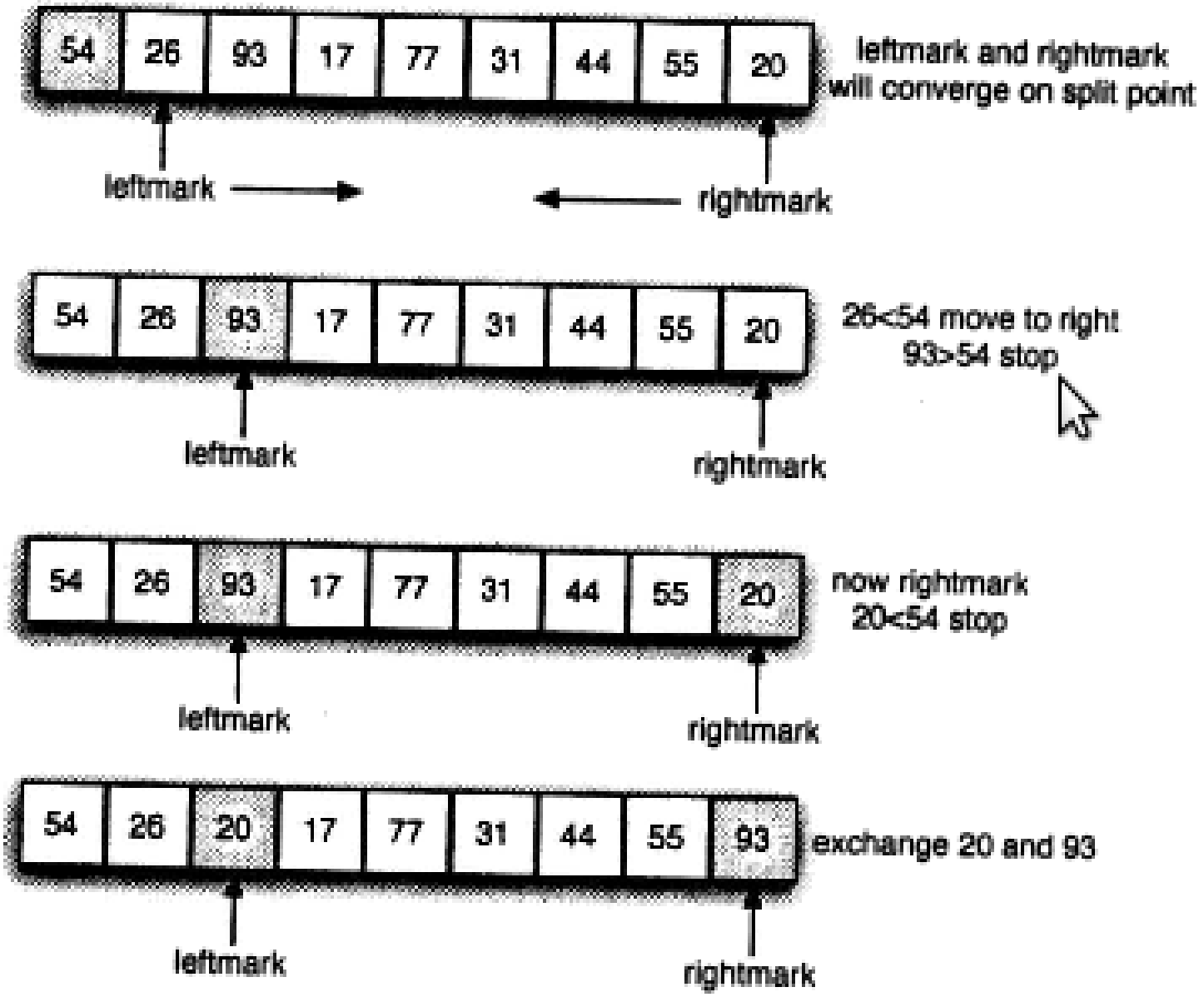
→ bu bölme için çok önemli

bölme noktasının bulunması

- → pivot (= 54) değerinin listenin sıralanmış halindeki yerini buluncaya kadar tara
- → soldan - leftmark (= 1), sağdan - rightmark (= 8)
- → pivot'tan büyük bir değere ulaşıncaya kadar leftmark'ı ilerlet (= 2, 93 > 54)
- → leftmark'taki değerden küçük bir değere (< 93) ulaşıncaya kadar, rightmark'i ilerlet (= 8)
- → bunları yer değiştir (93 <--> 20)

bölme noktasının bulunması

gösterim



bölme noktasının bulunması: dur

→ leftmark, rightmark'ı geçtiyse dur

→ bölme noktasını bulduk

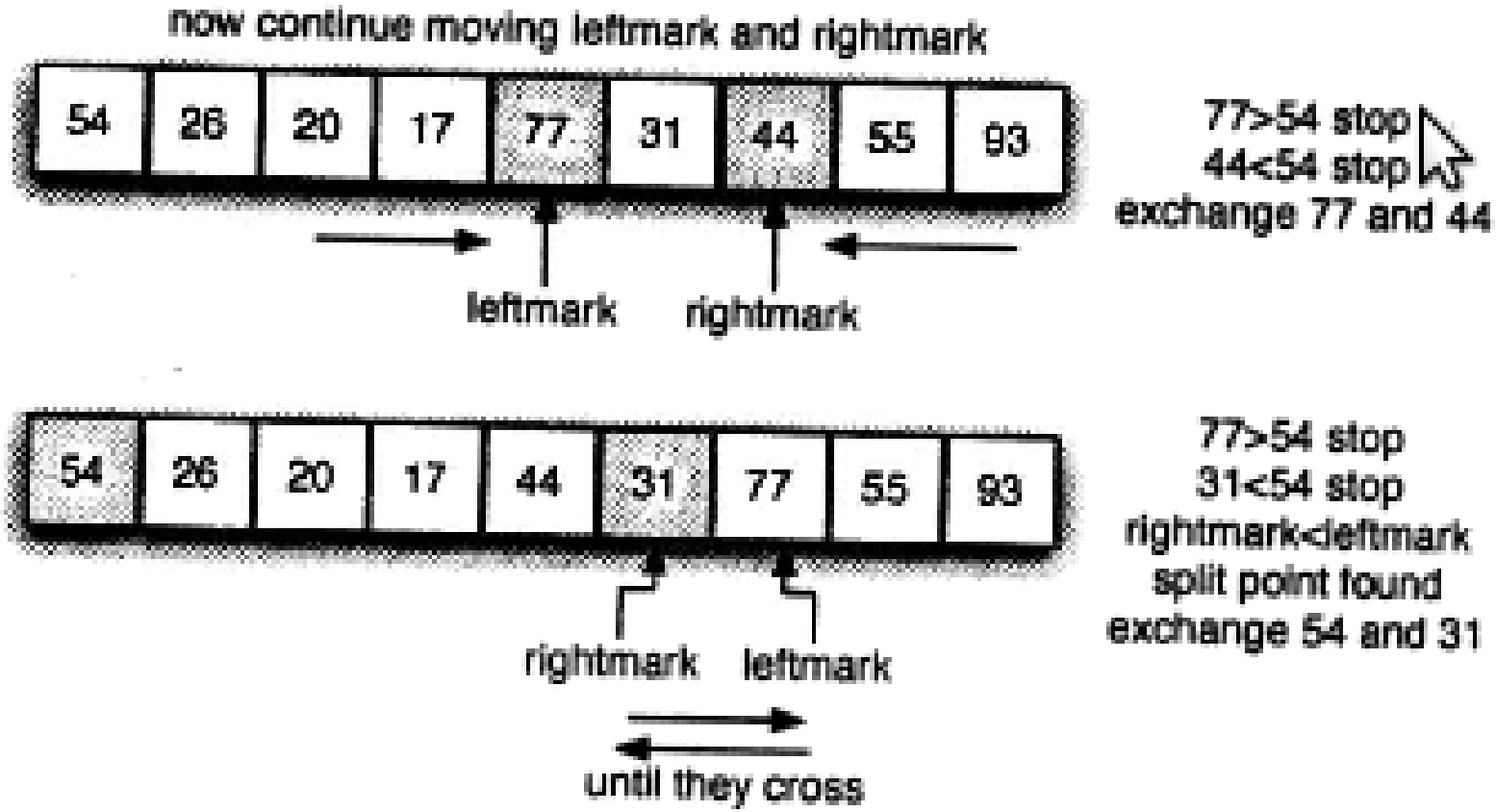


Figure 4.25: Finding the Split Point for 54

algoritma

bölme noktasının bulunması

a) listede **sağa** doğru giderken pivot'tan **büyük** olana ulaşınca dur (leftmark)

b) listede **sola** doğru giderken pivot'tan **küçük** olana ulaşınca dur (rightmark)

c) takas(leftmark, rightmark)

d) rightmark $<$ leftmark oluncaya kadar a–b–c işlerini tekrarla

bölme noktası $=$ rightmark

algoritma

böl ve yönet,

e) takas(rightmark, pivot)

→ solAltListe = liste[:rightmark], sağAltListe = liste[rightmark:]

→ solAltListe < pivot, sağAltListe > pivot

f) sağ|solAltListe'yi quicksort et

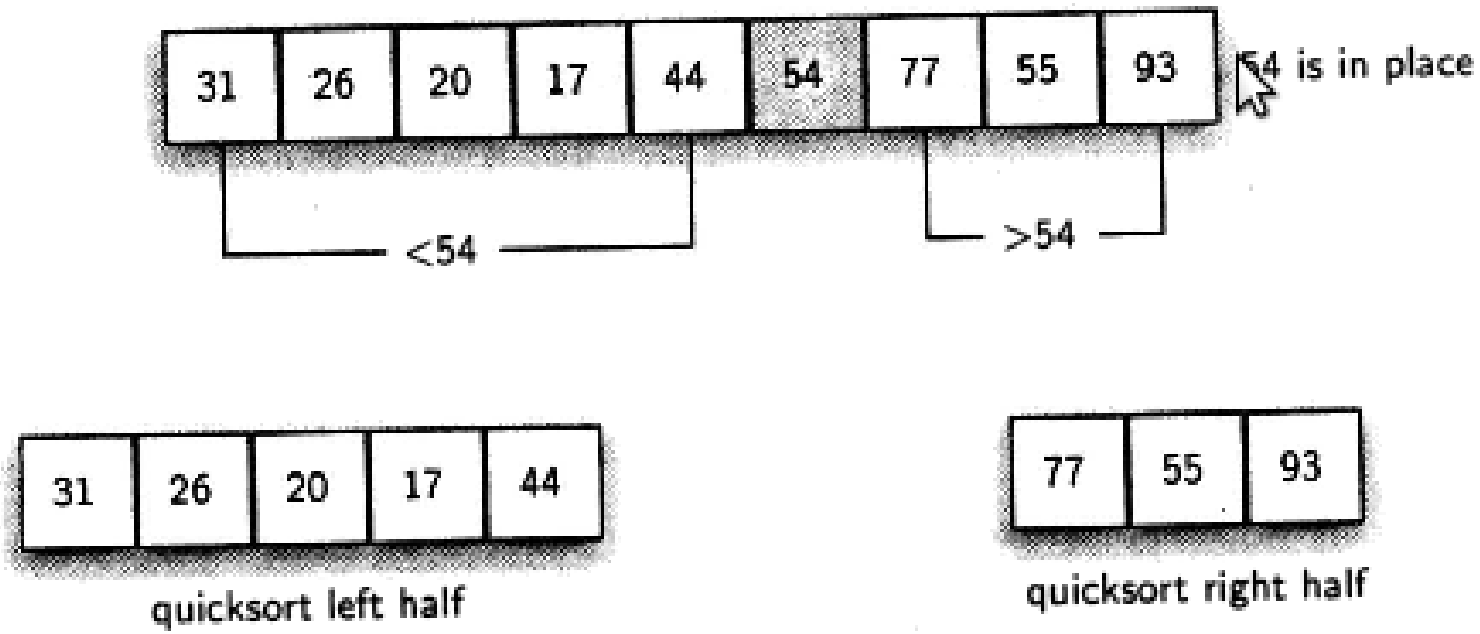


Figure 4.26: Completing the Partition Process to Find the Split Point for 54

gerçekleme: sort

→ gerçekleme: sort

```
1       def quickSort(alist):
2           quickSortHelper(alist,0,len(alist)-1)
3
4       def quickSortHelper(alist,first,last):
5           if first<last:
6
7               splitpoint = partition(alist,first,last)
8
9               quickSortHelper(alist,first,splitpoint-1)
10              quickSortHelper(alist,splitpoint+1,last)
```

gerçekleme: partition

→ gerçekleme: partition

```
1       def partition(alist,first,last):
2           pivotvalue = alist[first]
3
4           leftmark, rightmark = first+1, last
5
6           done = False
7           while not done:
8               while leftmark <= rightmark and \
9                       alist[leftmark] < pivotvalue:
10                  leftmark = leftmark + 1
11              while alist[rightmark] > pivotvalue and \
12                      rightmark >= leftmark:
13                  rightmark = rightmark -1
14              if rightmark < leftmark:
15                  done = True
16              else:
17                  alist[leftmark],alist[rightmark]= \
18                              alist[rightmark],alist[leftmark]
19
20          alist[first],alist[rightmark]= \
21                          alist[rightmark],alist[first]
22          return rightmark
```

analiz

- → sürekli olarak split point listenin ortasında olursa
- → log n kez bölme gerçekleşir
- → split pointi bulmak için n elemanın her bir idenetlenir
- → sonuç n log n
- → ekstra bellek gerektirmez

analiz: kötü

- → split point her zaman listenin ilk elemanı ise
- → length(solAltListe) = 0 ise
- → en kötü durum
- → length(sağAltListe) = n-1, n-2, ..., 1
- → dolayısıyla O(n^2)

pivot seçimi

- → pivot seçimi önemlidir
- → farklı yolları var
- → üç değerin ortancası: baş, orta, son
- → median(54, 77, 20) $\Longrightarrow$ **54**

→ liste sıralıya yakınsa çok iyi sonuç verir

→ ödev: quicksort gerçeklemesinde pivot seçimini değiştirin

4.5 özet

- → algoritma analizi gerçekleme bağımsızdır
- → Big-O bir gösterimdir, problem boyutu cinsinden baskın bileşenine göre sınıflandırır
- → sequential arama, sıralı/sırasız için O(n^2)
- → sıralı listede binary arama O(log n)
- → çırpı tabloları sabit zamanlı arama O(1)

özet: sıralama

- → bubble/selection/insertion sıralama O(n^2)
- → shell, insertion iyileştirir O(n) - O(n^2)
- → merge, O(n log n), ekstra bellek gerektirir
- → quicksort, O(n log n), ekstra bellek gerekmez
- → quicksort, split point listenin ortasına yakın çıkmazsa O(n^2)