Bölüm 10. Sıralama

Olcay Taner Yıldız

2014

Giriş

Sıralama Algoritmaları

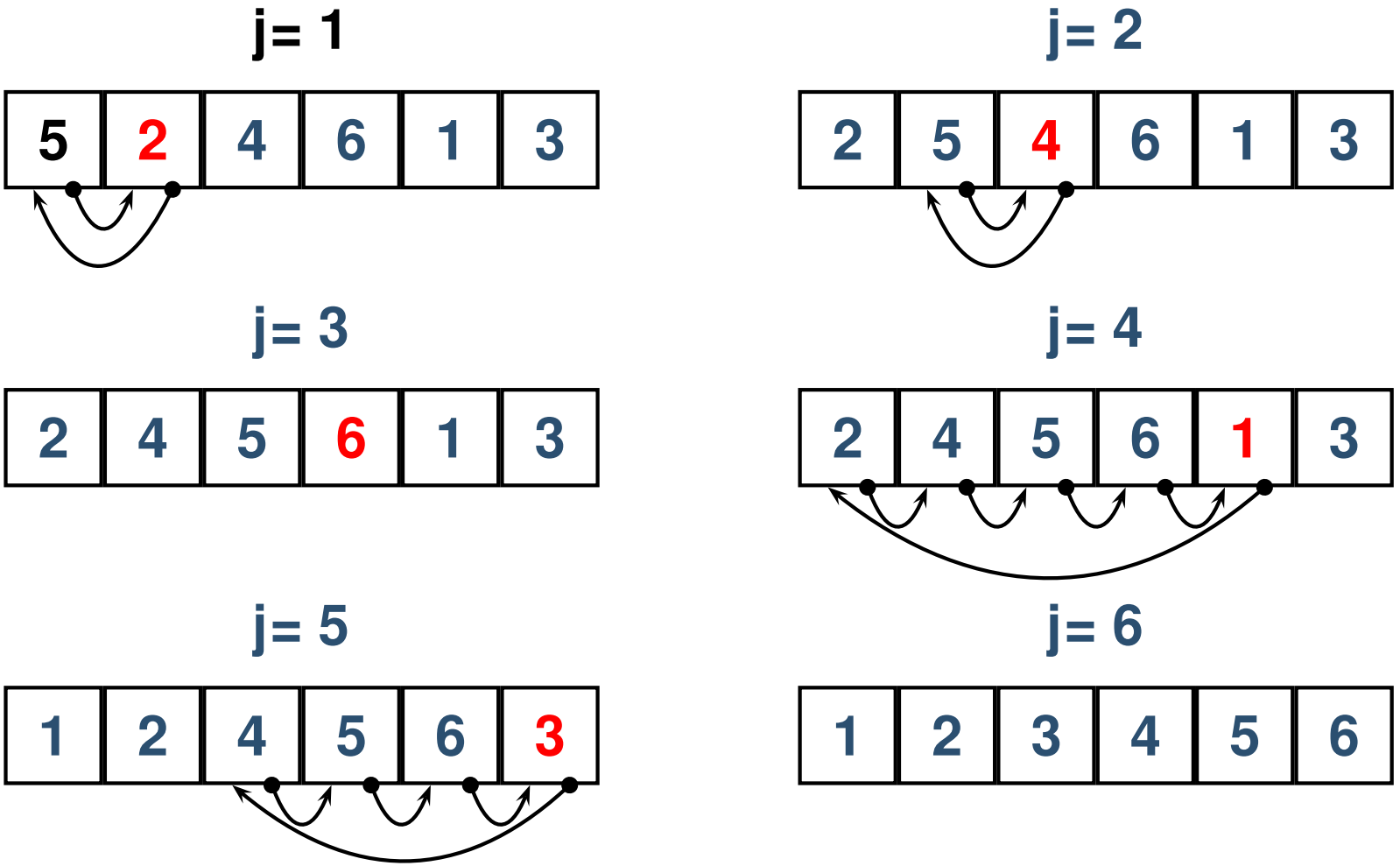
- Birinci kategorideki algoritmaların temel fikri ters sırada dizilmiş elemanları tekrar doğru sıralarına getirmek olarak özetlenebilir. Doğaları gereği ardışık elemanların karşılaştırılmasına dayalı bu algoritmaların çalışma süreleri ortalamada $\mathcal{O}(N^2)$'den az olamaz.
- İkinci kategoride daha karmaşık, belirli veri yapılarına veya algoritma prensiplerine dayanan hızlı sıralama algoritmaları vardır. Her üç sıralama algoritması da elemanları karşılaştırma prensibine dayandıkları için doğaları gereği çalışma süreleri ortalamada $\mathcal{O}(N \log N)$'den az olamaz.
- Üçüncü kategoride ise elemanları karşılaştırma prensibine dayanmayan, bu sebeple de sadece kesikli verilere uygulanabilen sıralama algoritmaları vardır. Bu algoritmaların çalışma süreleri olabilecek en kısa çalışma süresi olan $\mathcal{O}(N)$'den az olamaz.

Eklemeli Sıralama

Eklemeli sıralama algoritması


```
1  void eklemeli(int[] A){
2      int i, j, t;
3      for (j = 1; j < A.length; j++){
4          t = A[j];
5          i = j - 1;
6          while (i >= 0 && A[i] > t){
7              A[i+1] = A[i];
8              i = i - 1;
9          }
10         A[i + 1] = t;
11     }
12 }
```

Altı elemandan oluşan bir dizinin eklemeli sıralama yöntemiyle sıralanması



Seçmeli Sıralama

Seçmeli sıralama algoritması


```
1  void secmeli(int[] A){
2      int i, j, pos, min;
3      for (i = 0; i < A.length - 1; i++){
4          min = A[i];
5          pos = i;
6          for (j = i + 1; j < A.length; j++)
7              if (A[j] < min){
8                  min = A[j];
9                  pos = j;
10             }
11         if (pos != i){
12             A[pos] = A[i];
13             A[i] = min;
14         }
15     }
16 }
```

Altı elemandan oluşan bir dizinin seçmeli sıralama yöntemiyle sıralanması

i= 0

5	2	4	6	1	3

i= 1

1	2	4	6	5	3

i= 2

1	2	4	6	5	3

i= 3

1	2	3	6	5	4

i= 4

1	2	3	4	5	6

i= 5

1	2	3	4	5	6

Kabarcık Sıralama

Kabarcık sıralama algoritması


```
1  void kabarcik(int[] A){
2      int i, tmp;
3      boolean degistirdi = true;
4      while (degistirdi){
5          degistirdi = false;
6          for (i = 0; i < A.length - 1; i++)
7              if (A[i] > A[i + 1]){
8                  degistirdi = true;
9                  tmp = A[i];
10                 A[i] = A[i + 1];
11                 A[i + 1] = tmp;
12             }
13     }
14 }
```

Altı elemandan oluşan bir dizinin kabarcık sıralama yöntemiyle sıralanması

j= 1

5	2	4	6	1	3

j= 2

2	4	5	1	3	6

j= 3

2	4	1	3	5	6

j= 4

2	1	3	4	5	6

j= 5

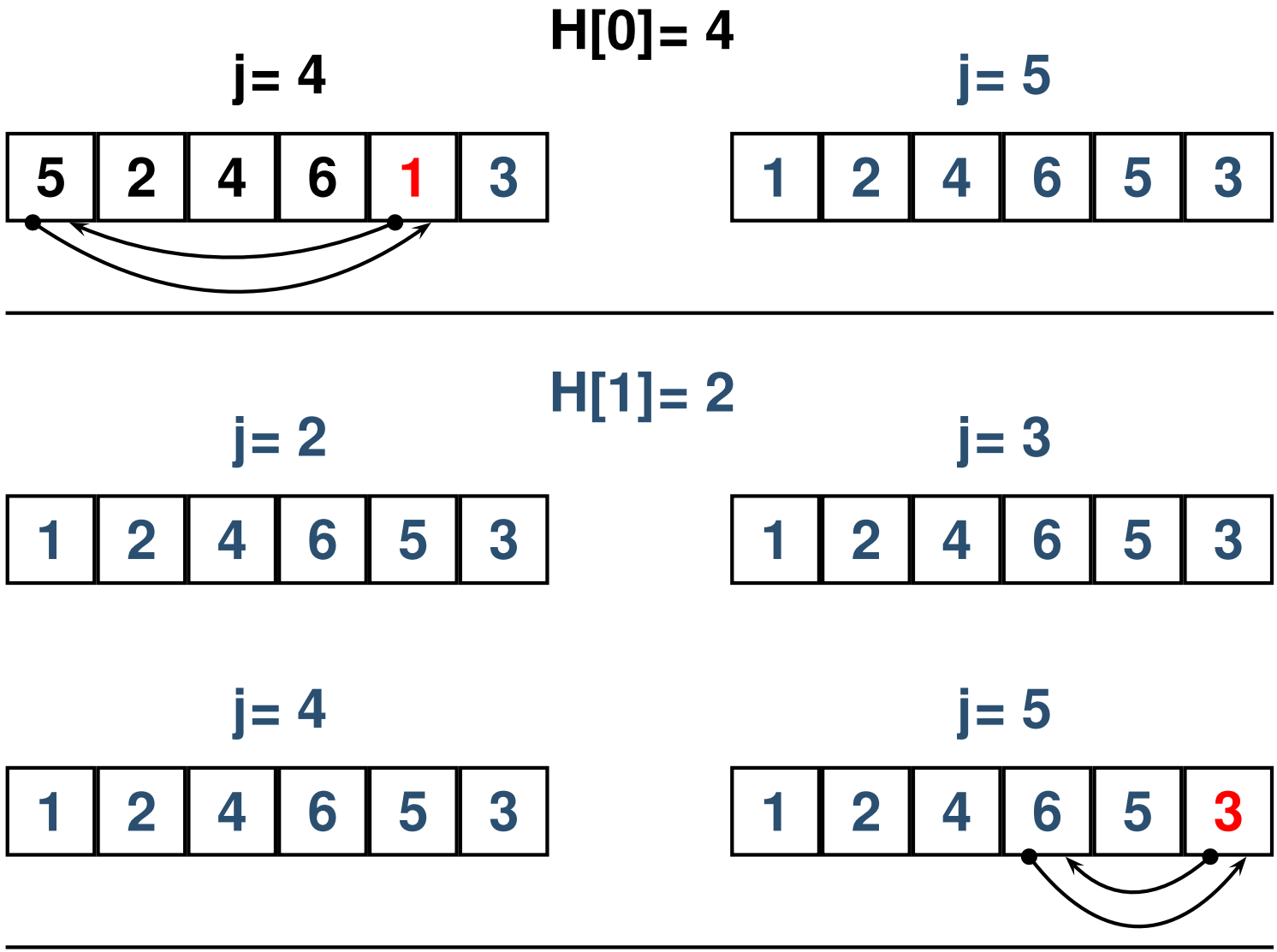
1	2	3	4	5	6

Shell Sıralama

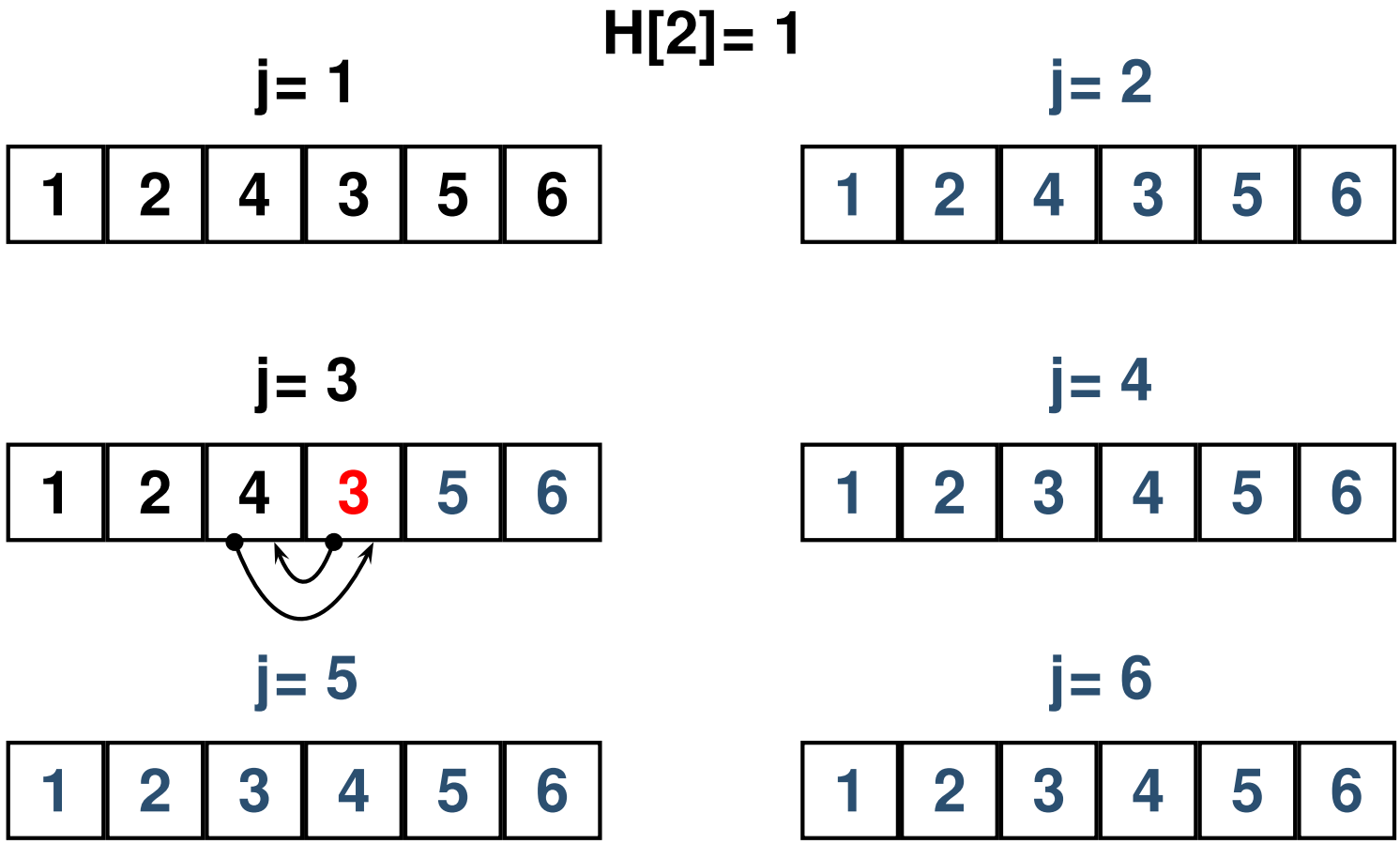
Shell sıralama algoritması


```
1  void shell(int[] A, int[] H){
2      int i, j, t, k, fark;
3      for (k = 0; k < H.length; k++){
4          fark = H[k];
5          for (j = fark; j < A.length; j++){
6              t = A[j];
7              i = j - fark;
8              while (i >= 0 && A[i] > t){
9                  A[i+fark] = A[i];
10                 i = i - fark;
11             }
12             A[i + fark] = t;
13         }
14     }
15 }
```

Altı elemandan oluşan bir dizinin Shell sıralama yöntemiyle sıralanması (1)



Altı elemandan oluşan bir dizinin Shell sıralama yöntemiyle sıralanması (2)

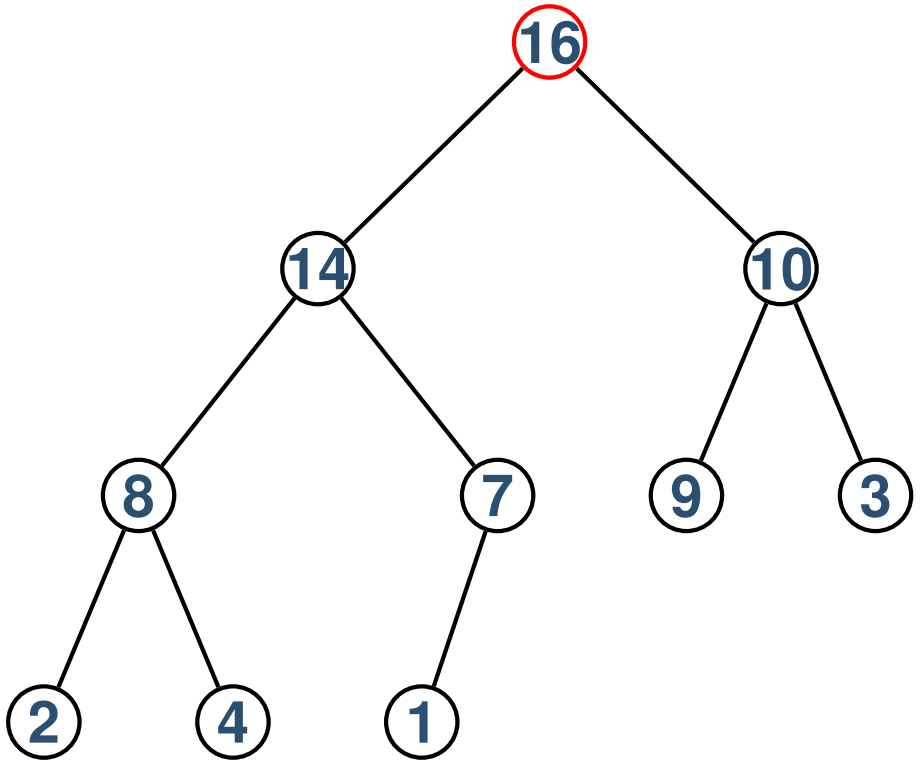


Yığın Sıralama

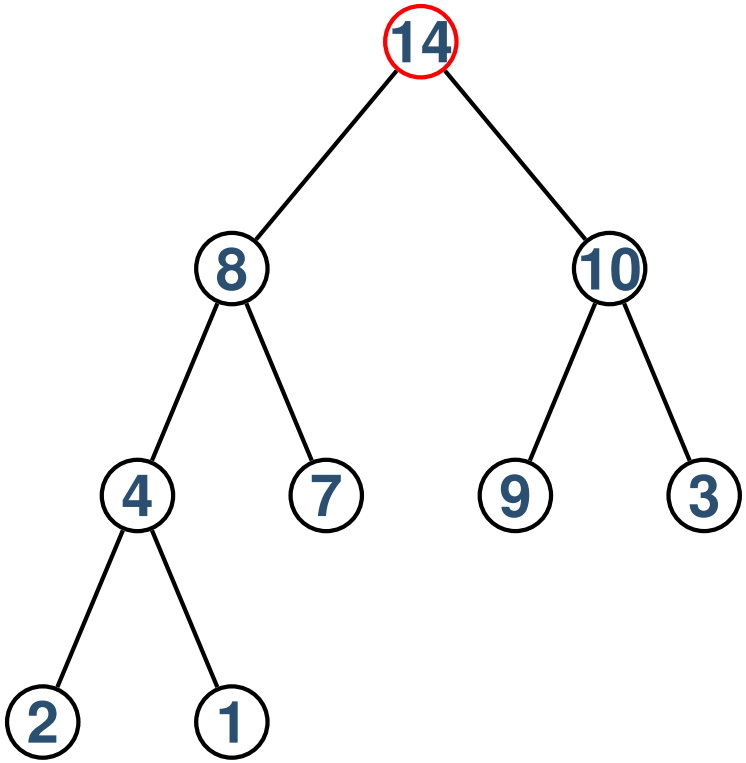
Yığın sıralama algoritması


```
1  void yiginSiralama(int[] A){
2      int i;
3      Nokta e;
4      Yigin y;
5      y = new Yigin(A.length);
6      for (i = 0; i < A.length; i++){
7          e = new Nokta(A[i], A[i]);
8          y.yiginEkle(e);
9      }
10     for (i = 0; i < A.length; i++){
11         e = y.azamiDondur();
12         A[i] = e.icerik;
13     }
14 }
```

Örnek bir dizinin yığın sıralama algoritmasıyla sıralanması (1)

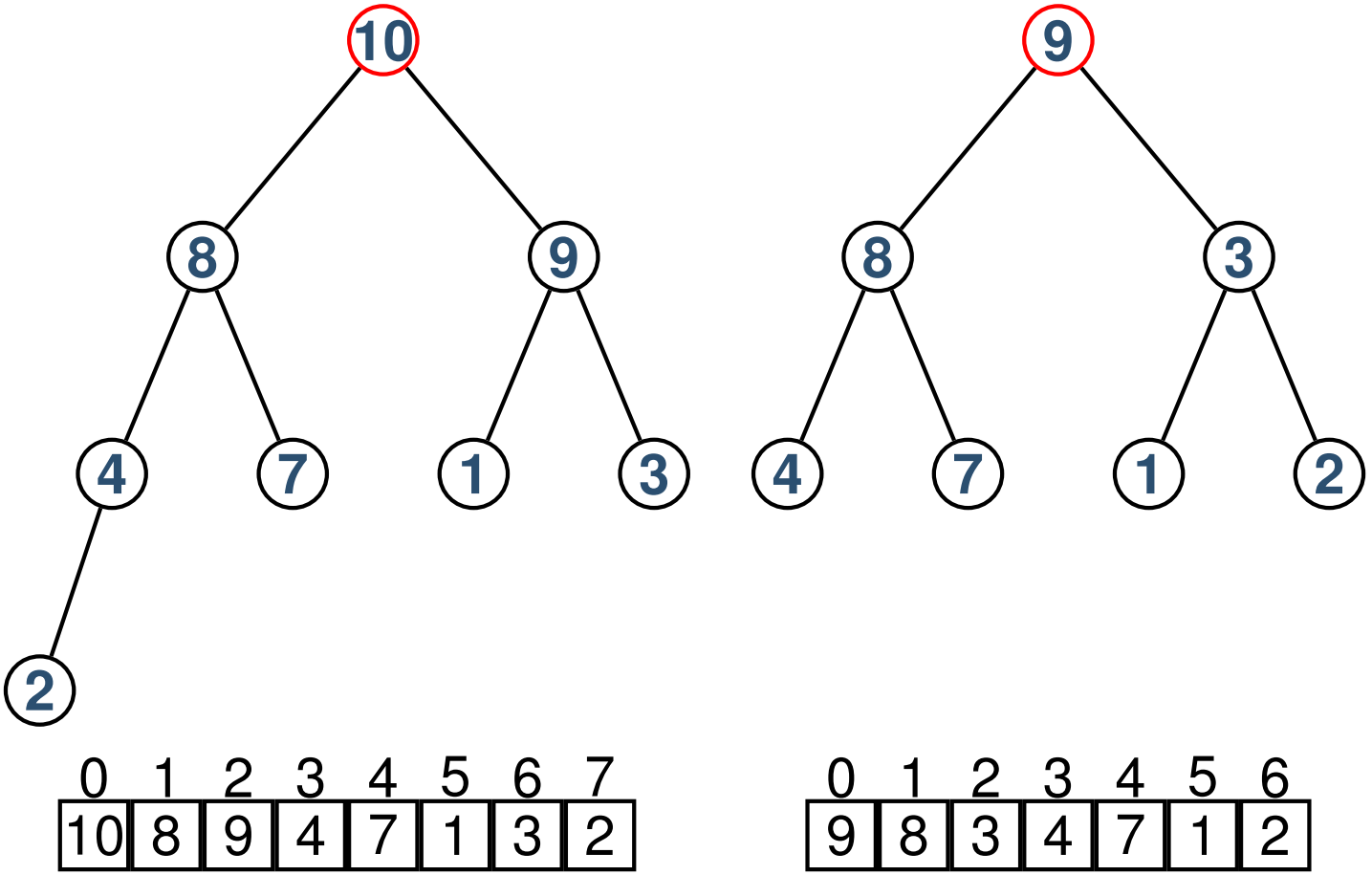


0	1	2	3	4	5	6	7	8	9
16	14	10	8	7	9	3	2	4	1

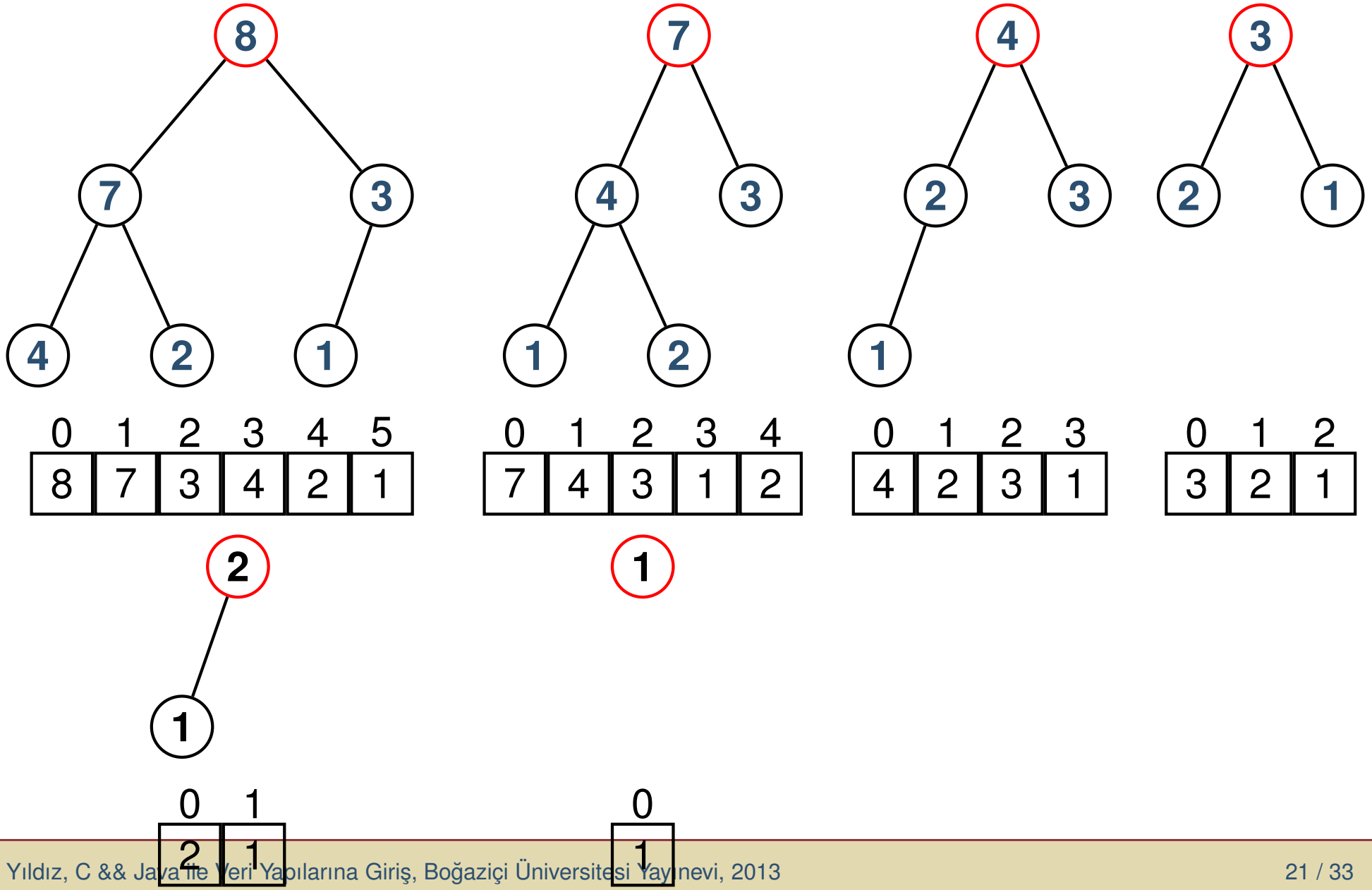


0	1	2	3	4	5	6	7	8
14	8	10	4	7	9	3	2	1

Örnek bir dizinin yığın sıralama algoritmasıyla sıralanması (2)



Örnek bir dizinin yığın sıralama algoritmasıyla sıralanması (3)



Birleştirmeli Sıralama

Birleştirmeli sıralama algoritması (1)


```
1  void birlestir (int[] A, int p, int q, int r){
2      int L[], R[];
3      int n1, n2, i, j, k;
4      n1 = q - p + 1;
5      n2 = r - q;
6      L = new int[n1 + 1];
7      R = new int[n2 + 1];
8      for (i = 0; i < n1; i++)
9          L[i] = A[p + i];
10     for (i = 0; i < n2; i++)
11         R[i] = A[q + i];
12     L[n1] = Integer.MAX_VALUE;
13     R[n2] = Integer.MAX_VALUE;
14     i = 0;
15     j = 0;
```

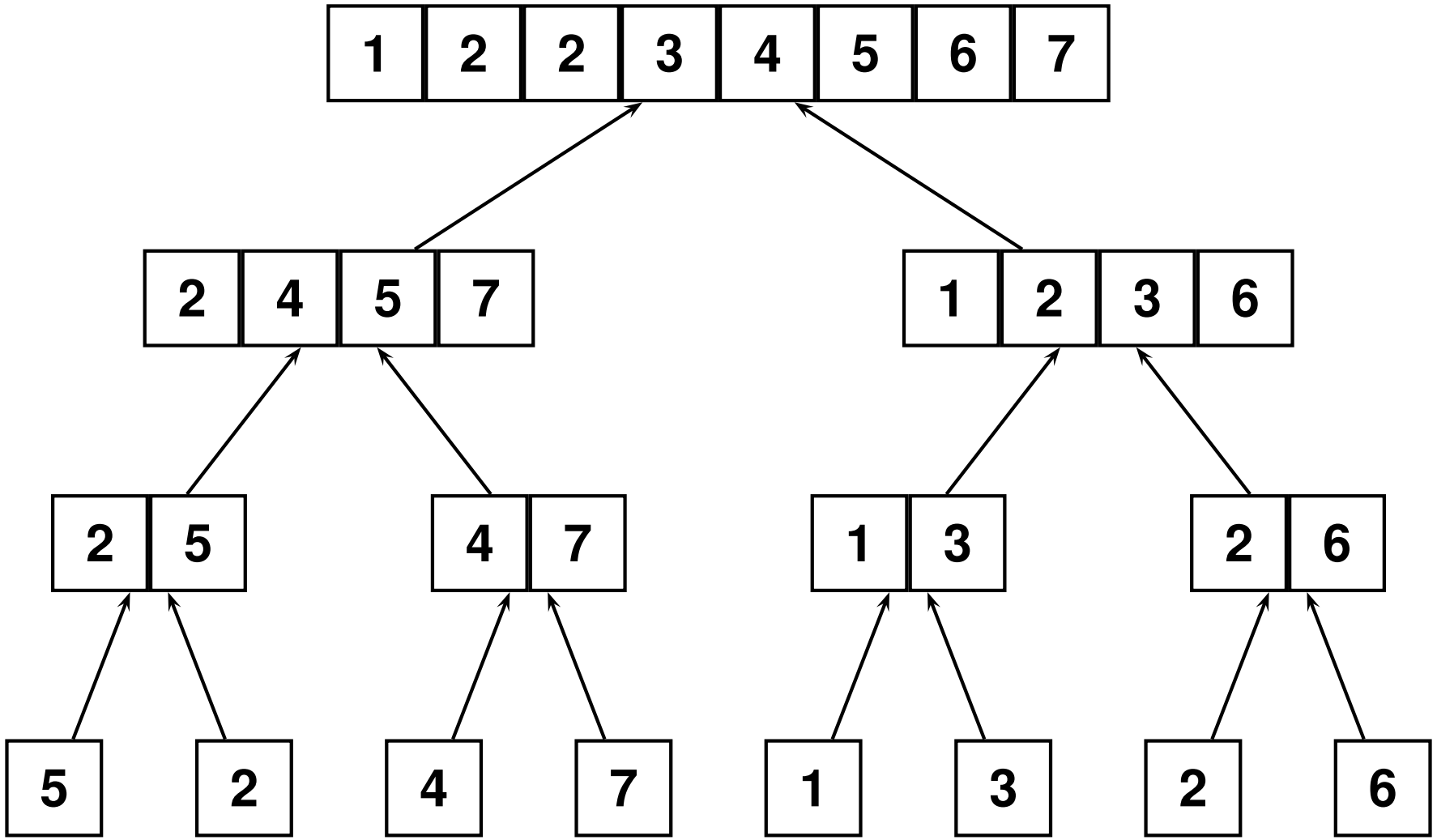
Birleştirmeli sıralama algoritması (2)


```
16      for (k = p; k <= r; k++)
17          if (L[i] <= R[j]){
18              A[k] = L[i];
19              i++;
20          }
21          else{
22              A[k] = R[j];
23              j++;
24          }
25  }
26  void birlestirmeli (int[] A, int bas, int son){
27      int pivot;
28      if (bas < son){
29          pivot = (bas + son) / 2;
30           birlestirmeli (A, bas, pivot);
31           birlestirmeli (A, pivot+1, son);
32           birlestir (A, bas, pivot, son);
33      }
34  }
```

4 elemanlı sıralı iki dizinin birleştirilmesi

2 4 5 7 -	1 2 3 6 -	1
2 4 5 7 -	1 2 3 6 -	1 2
2 4 5 7 -	1 2 3 6 -	1 2 2
2 4 5 7 -	1 2 3 6 -	1 2 2 3
2 4 5 7 -	1 2 3 6 -	1 2 2 3 4
2 4 5 7 -	1 2 3 6 -	1 2 2 3 4 5
2 4 5 7 -	1 2 3 6 -	1 2 2 3 4 5 6
2 4 5 7 -	1 2 3 6 -	1 2 2 3 4 5 6 7
2 4 5 7 -	1 2 3 6 -	1 2 2 3 4 5 6 7

8 elemandan oluşan bir dizinin Birleştirmeli Sıralama yöntemiyle sıralanması



Hızlı Sıralama

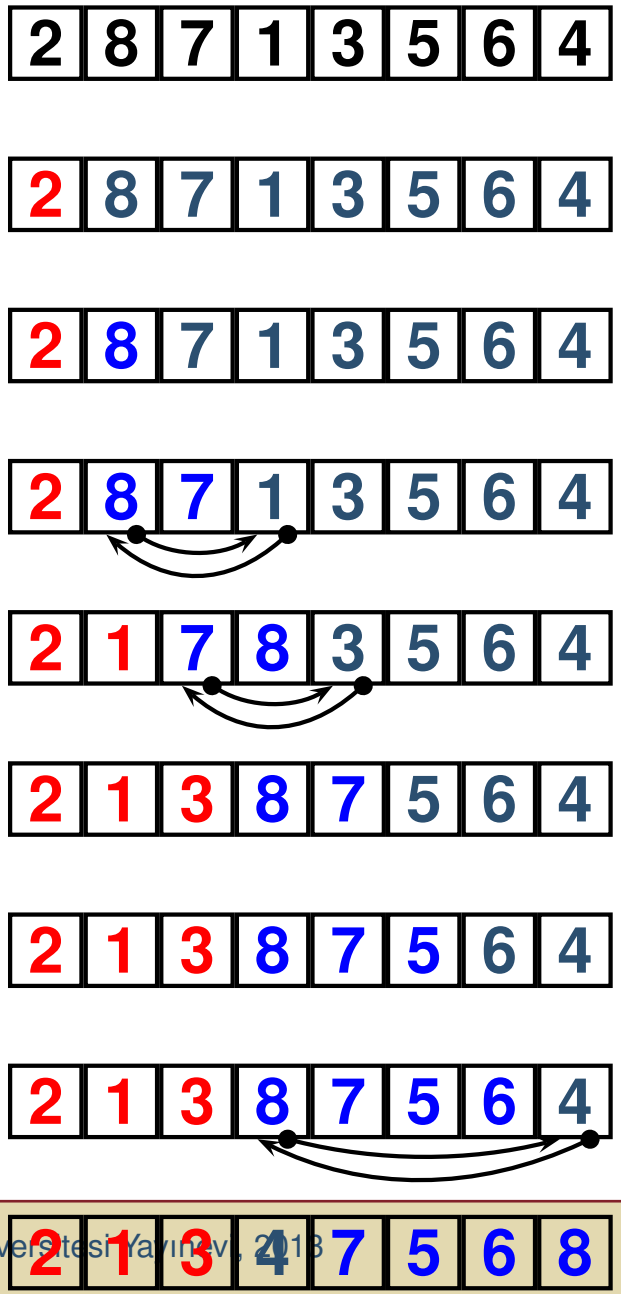
Hızlı sıralama algoritması


```
1 void hizli(int[] A, int bas, int son){
2     int pivot;
3     if (bas < son){
4         pivot = parcala(A, bas, son);
5         hizli (A, bas, pivot - 1);
6         hizli (A, pivot + 1, son);
7     }
8 }
```

Dizinin parçalanması


```
1  void yerDegis(int[] A, int i, int j){
2      int tmp;
3      tmp = A[i ];
4      A[i ]  = A[j ];
5      A[j ]  = tmp;
6  }
7  int parcala(int[] A, int bas, int son){
8      int x = A[son], tmp;
9      int i = bas -  1,  j ;
10     for (j  = bas;  j  < son;  j++)
11         if (A[j ]  <= x){
12             i++;
13             yerDegis(A, i ,  j );
14         }
15     yerDegis(A, i  + 1,  son);
16     return i + 1;
17 }
```

parcala fonksiyonunun örnek bir uygulanışı



Saymalı Sıralama

Saymalı sıralama algoritması


```
1  void saymali(int[] A, int k){
2      int i, n, C[], B[];
3      n = A.length;
4      C = new int[k];
5      B = new int[n];
6      for (i = 0; i < n; i++)
7          C[A[i]]++;
8      for (i = 1; i < k; i++)
9          C[i] = C[i] + C[i-1];
10     for (i = n - 1; i >= 0; i --){
11         B[C[A[i]] - 1] = A[i];
12         C[A[i]]--;
13     }
14     for (i = 0; i < n; i++)
15         A[i] = B[i];
16 }
```

Saymalı sıralama algoritmasının sekiz elemanlı bir dizi üzerinde gösterilmesi