

分而治之篇：逆序对计数问题

童咏昕

北京航空航天大学
计算机学院

中国大学MOOC北航《算法设计与分析》

	1	2	3	4	5
A	4	6	8	3	5

- 逆序对
 - 当 $i < j$ 时 $A[i] > A[j]$ 的二元组 $(A[i], A[j])$

问题背景：逆序对



	1	2	3	4	5
A	4	6	8	3	5

$1 < 4$ 且 $A[1] > A[4]$

- 逆序对
 - 当 $i < j$ 时 $A[i] > A[j]$ 的二元组 $(A[i], A[j])$
- 例如： $(A[1], A[4])$

问题背景： 逆序对



	1	2	3	4	5
A	4	6	8	3	5

$2 < 4$ 且 $A[2] > A[4]$

- 逆序对
 - 当 $i < j$ 时 $A[i] > A[j]$ 的二元组 $(A[i], A[j])$
- 例如: $(A[1], A[4])$, $(A[2], A[4])$

	1	2	3	4	5
A	4	6	8	3	5

- 逆序对
 - 当 $i < j$ 时 $A[i] > A[j]$ 的二元组 $(A[i], A[j])$
- 例如： $(A[1], A[4])$, $(A[2], A[4])$

问题： 数组A中共有多少个逆序对？

	1	2	3	4	5
A	4	6	8	3	5

- 逆序对
 - 当 $i < j$ 时 $A[i] > A[j]$ 的二元组 $(A[i], A[j])$
- 例如： $(A[1], A[4])$, $(A[2], A[4])$

问题： 数组 A 中共有多少个逆序对？

答案： 共5个， $\{(A[1], A[4]), (A[2], A[4]), (A[2], A[5]), (A[3], A[4]), (A[3], A[5])\}$

- 形式化定义

逆序对计数问题

Counting Inversions

输入

- 长度为 n 的数组 $A[1..n]$

- 形式化定义

逆序对计数问题

Counting Inversions

输入

- 长度为 n 的数组 $A[1..n]$

输出

- 数组 $A[1..n]$ 逆序对的总数，即：

$$\sum_{1 \leq i < j \leq n} X_{i,j}$$
$$X_{i,j} = \begin{cases} 1, & A[i] > A[j] \\ 0, & A[i] \leq A[j] \end{cases}$$

- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|----|---|----|---|----|----|----|----|---|----|----|----|
| A | 13 | 8 | 10 | 6 | 15 | 18 | 12 | 20 | 9 | 14 | 17 | 19 |

[illegible]

- [illegible]

- [illegible]

- [illegible]

- Diagram illustrating the counting of inversions for the element 15 in the array A. The array A is [13, 8, 10, 6, 15, 18, 12, 20, 9, 14, 17, 19]. Elements 8, 10, and 6 are highlighted in green. Arrows point from the labels $i = 1$ and $j = 5$ to the elements 13 and 15 respectively. Below the array, a row labeled "逆序对数" (Number of Inversions) shows the count for each element: 3, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?. The first cell contains the red number 3, which is the number of inversions for the element at index 5 (15).

- Diagram illustrating the counting of inversions in an array A .

Array A (Indices 1 to 12):

Index	1	2	3	4	5	6	7	8	9	10	11	12
Value	13	8	10	6	15	18	12	20	9	14	17	19

Arrows indicate indices $i = 1$ (pointing to 13) and $j = 6$ (pointing to 18).

Below the array, a row labeled "逆序对数" (Number of Inversions) shows the first cell containing the value 3, and the remaining cells containing question marks, indicating the total count of inversions is 3.

- [illegible]

- Diagram illustrating the counting of inversions in an array A . The array A contains 12 elements: 13, 8, 10, 6, 15, 18, 12, 20, 9, 14, 17, 19. Elements 8, 10, 6, and 12 are highlighted in green. Below the array, a row labeled "逆序对数" (Number of Inversions) shows the first cell containing the value 4, and the remaining cells containing question marks. Arrows point from the labels $i = 1$ and $j = 8$ to the first and eighth elements of the array, respectively.

- Diagram illustrating the counting of inversions in an array A .

Array A (Indices 1 to 12):

Index	1	2	3	4	5	6	7	8	9	10	11	12
Value	13	8	10	6	15	18	12	20	9	14	17	19

Elements 8, 10, 6, 12, and 9 are highlighted in green.

Below the array, a row labeled "逆序对数" (Number of Inversions) shows the first cell containing the value 5, and the remaining cells containing question marks.

Arrows indicate the indices $i = 1$ and $j = 9$ corresponding to the first and ninth elements of the array.

- Diagram illustrating the counting of inversions in an array A . The array A contains 12 elements: 13, 8, 10, 6, 15, 18, 12, 20, 9, 14, 17, 19. Elements 8, 10, 6, 12, and 9 are highlighted in green. Below the array, a row labeled "逆序对数" (Number of Inversions) shows the first cell containing the value 5, and the remaining cells containing question marks. Arrows point from the first and last elements of the array to the first and last cells of the inversion count row, labeled $i = 1$ and $j = 12$ respectively.

- 对于数组的每个元素 $A[i]$, 枚举 $j(j > i)$, 并统计逆序对数目

	1	2	3	4	5	6	7	8	9	10	11	12
A	13	8	10	6	15	18	12	20	9	14	17	19

逆序对数	5	1	2	0	3	4	1	4	0	0	0	0
------	---	---	---	---	---	---	---	---	---	---	---	---

蛮力枚举：伪代码



- 对于数组的每个元素 $A[i]$ ，枚举 $j(j > i)$ ，并统计逆序对数目

```
输入: 一个大小为 $n$ 的数组 $A[1..n]$   
输出: 数组 $A[1..n]$ 中逆序对数目 $Sum$   
 $Sum \leftarrow 0$   
for  $i \leftarrow 1$  to  $n$  do  
    for  $j \leftarrow i + 1$  to  $n$  do  
        if  $A[i] > A[j]$  then  
             $Sum \leftarrow Sum + 1$   
        end  
    end  
end  
return  $Sum$ 
```

初始化逆序对数目

蛮力枚举：伪代码



- 对于数组的每个元素 $A[i]$ ，枚举 $j(j > i)$ ，并统计逆序对数目

```
输入: 一个大小为 $n$ 的数组 $A[1..n]$   
输出: 数组 $A[1..n]$ 中逆序对数目 $Sum$   
 $Sum \leftarrow 0$   
for  $i \leftarrow 1$  to  $n$  do  
    for  $j \leftarrow i + 1$  to  $n$  do  
        if  $A[i] > A[j]$  then  
             $Sum \leftarrow Sum + 1$   
        end  
    end  
end  
return  $Sum$ 
```

枚举所有下标 i

蛮力枚举：伪代码



- 对于数组的每个元素 $A[i]$ ，枚举 $j(j > i)$ ，并统计逆序对数目

```
输入: 一个大小为 $n$ 的数组 $A[1..n]$   
输出: 数组 $A[1..n]$ 中逆序对数目 $Sum$   
 $Sum \leftarrow 0$   
for  $i \leftarrow 1$  to  $n$  do  
    for  $j \leftarrow i + 1$  to  $n$  do  
        if  $A[i] > A[j]$  then  
             $Sum \leftarrow Sum + 1$   
        end  
    end  
end  
return  $Sum$ 
```

枚举右侧元素 j
统计逆序对数目



蛮力枚举：伪代码

- 对于数组的每个元素 $A[i]$ ，枚举 $j(j > i)$ ，并统计逆序对数目

```
输入: 一个大小为 $n$ 的数组 $A[1..n]$   
输出: 数组 $A[1..n]$ 中逆序对数目 $Sum$   
 $Sum \leftarrow 0$   
for  $i \leftarrow 1$  to  $n$  do  
    for  $j \leftarrow i + 1$  to  $n$  do  
        if  $A[i] > A[j]$  then  
             $Sum \leftarrow Sum + 1$   
        end  
    end  
end  
return  $Sum$ 
```

时间复杂度: $O(n^2)$

$O(n)$ $O(n^2)$

蛮力枚举：伪代码



- 对于数组的每个元素 $A[i]$ ，枚举 $j(j > i)$ ，并统计逆序对数目

```
输入: 一个大小为 $n$ 的数组 $A[1..n]$   
输出: 数组 $A[1..n]$ 中逆序对数目 $Sum$   
 $Sum \leftarrow 0$   
for  $i \leftarrow 1$  to  $n$  do  
    for  $j \leftarrow i + 1$  to  $n$  do  
        if  $A[i] > A[j]$  then  
             $Sum \leftarrow Sum + 1$   
        end  
    end  
end  
return  $Sum$ 
```

时间复杂度: $O(n^2)$

$O(n)$ $O(n^2)$

问题：如何进一步提高算法效率？能否采用分而治之策略？

分而治之：一般步骤



分解原问题

原问题分解成多个子问题



解决子问题

递归地求解各个子问题



合并问题解

将结果合并为原问题解

分而治之：算法框架



$A[1..n]$

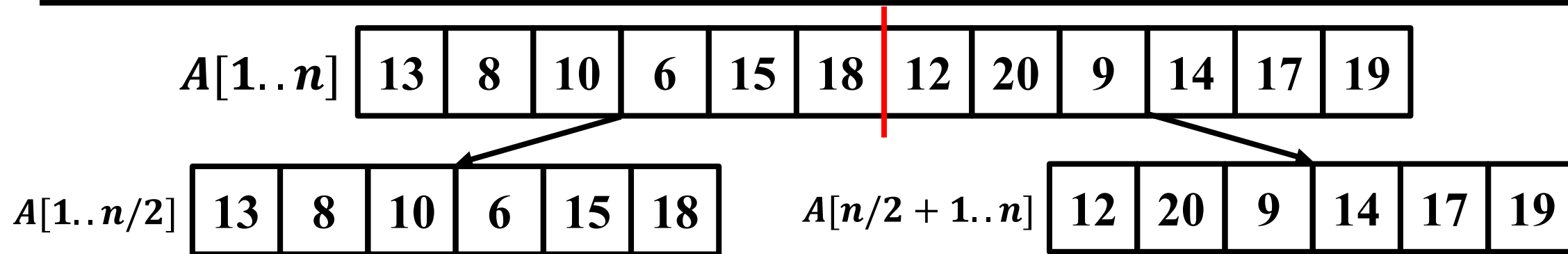
13	8	10	6	15	18	12	20	9	14	17	19
----	---	----	---	----	----	----	----	---	----	----	----

分解原问题

解决子问题

合并问题解

分而治之：算法框架



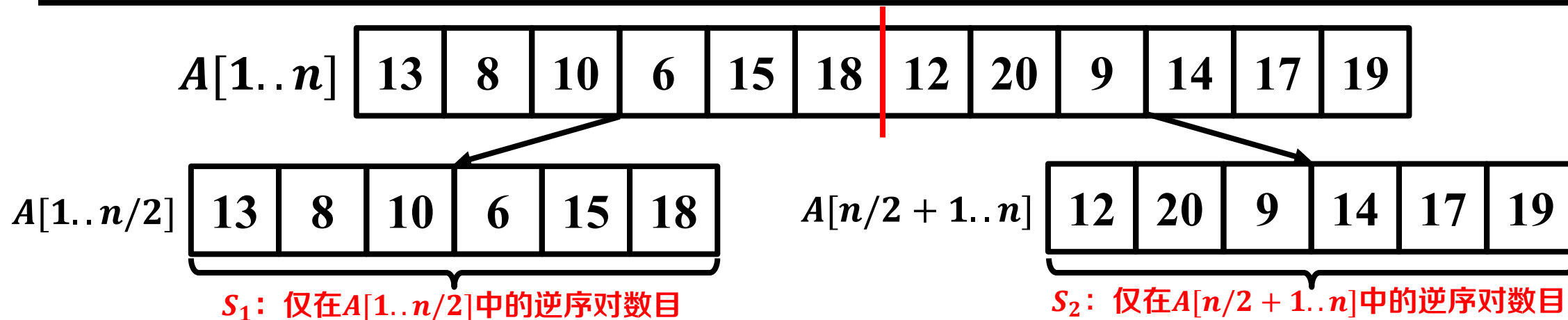
- 把数组 A 二分为两个子数组 $A[1..n/2]$, $A[n/2 + 1..n]$

分解原问题

解决子问题

合并问题解

分而治之：算法框架



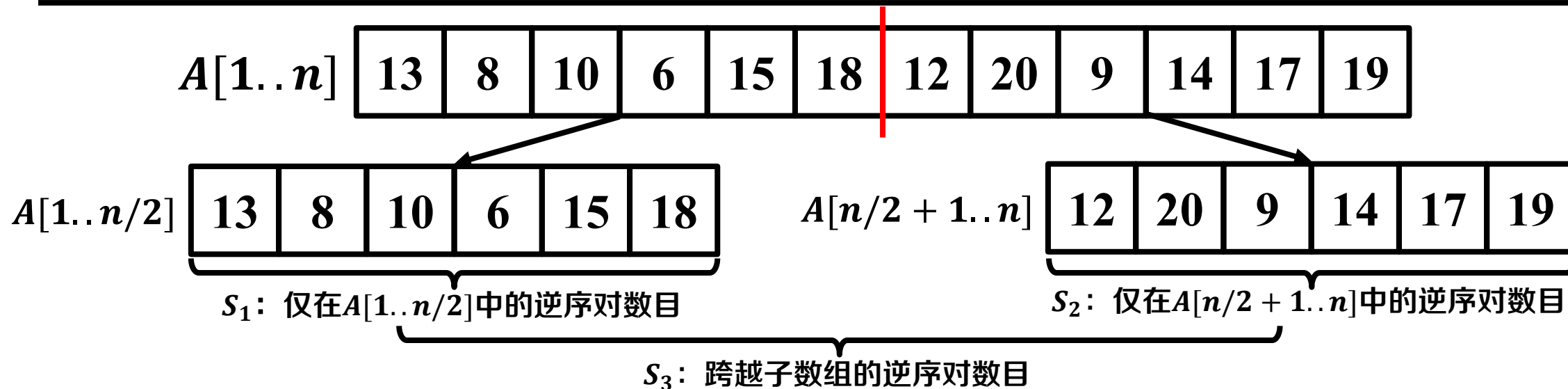
- 把数组 A 二分为两个子数组 $A[1..n/2]$, $A[n/2 + 1..n]$
- 递归求解子问题
 - 求解 S_1 : 仅在 $A[1..n/2]$ 中的逆序对数目
 - 求解 S_2 : 仅在 $A[n/2 + 1..n]$ 中的逆序对数目

分解原问题

解决子问题

合并问题解

分而治之：算法框架



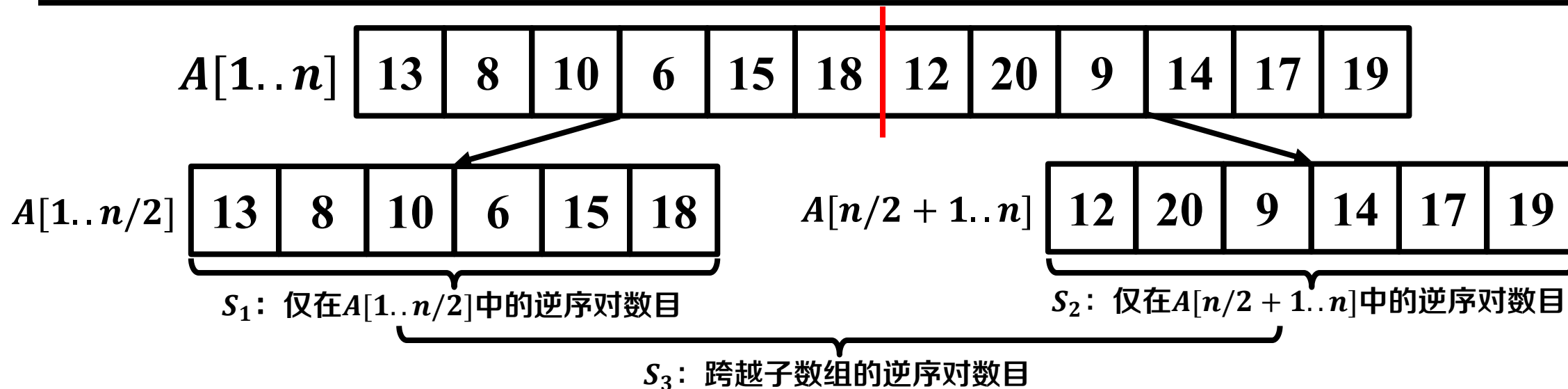
- 把数组 A 二分为两个子数组 $A[1..n/2]$, $A[n/2 + 1..n]$
- 递归求解子问题
 - 求解 S_1 : 仅在 $A[1..n/2]$ 中的逆序对数目
 - 求解 S_2 : 仅在 $A[n/2 + 1..n]$ 中的逆序对数目
- 合并 $A[1..n/2]$ 和 $A[n/2 + 1..n]$ 的解
 - 求解 S_3 : 跨越子数组的逆序对数目

分解原问题

解决子问题

合并问题解

分而治之：算法框架



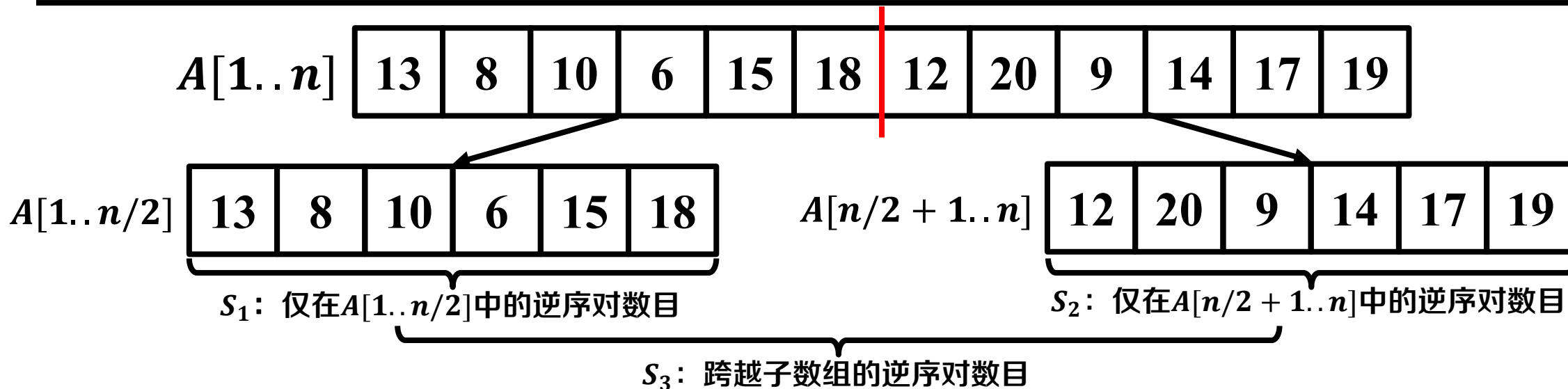
- 把数组 A 二分为两个子数组 $A[1..n/2]$, $A[n/2 + 1..n]$
- 递归求解子问题
 - 求解 S_1 : 仅在 $A[1..n/2]$ 中的逆序对数目
 - 求解 S_2 : 仅在 $A[n/2 + 1..n]$ 中的逆序对数目
- 合并 $A[1..n/2]$ 和 $A[n/2 + 1..n]$ 的解
 - 求解 S_3 : 跨越子数组的逆序对数目
 - $S = S_1 + S_2 + S_3$

分解原问题

解决子问题

合并问题解

分而治之：算法框架



- 把数组 A 二分为两个子数组 $A[1..n/2]$, $A[n/2 + 1..n]$

分解原问题

- 递归求解子问题

- 求解 S_1 : 仅在 $A[1..n/2]$ 中的逆序对数目
- 求解 S_2 : 仅在 $A[n/2 + 1..n]$ 中的逆序对数目

可递归求解

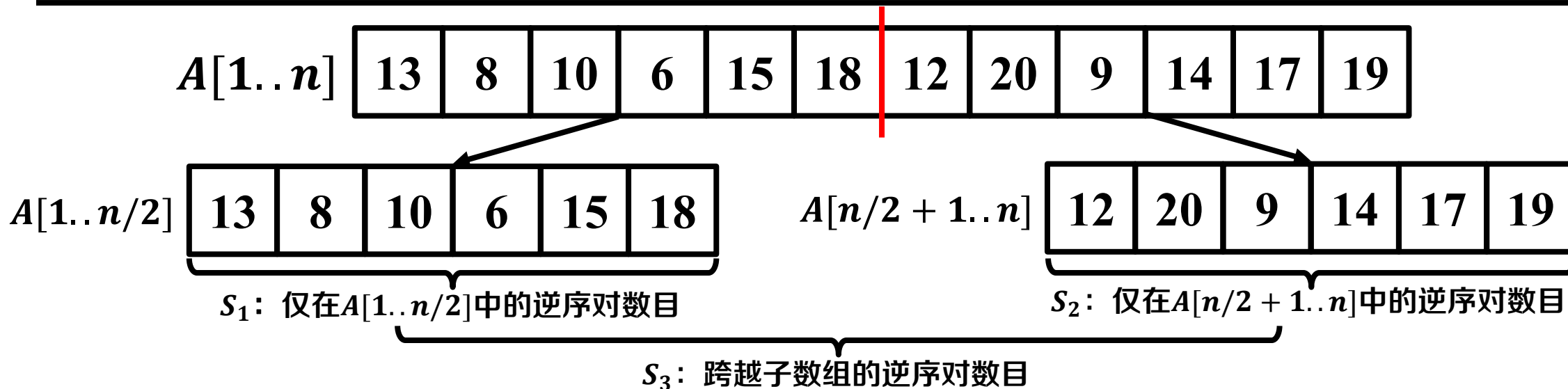
解决子问题

- 合并 $A[1..n/2]$ 和 $A[n/2 + 1..n]$ 的解

- 求解 S_3 : 跨越子数组的逆序对数目
- $S = S_1 + S_2 + S_3$

合并问题解

分而治之：算法框架



- 把数组 A 二分为两个子数组 $A[1..n/2]$, $A[n/2 + 1..n]$

分解原问题

- 递归求解子问题

- 求解 S_1 : 仅在 $A[1..n/2]$ 中的逆序对数目
- 求解 S_2 : 仅在 $A[n/2 + 1..n]$ 中的逆序对数目

可递归求解

解决子问题

- 合并 $A[1..n/2]$ 和 $A[n/2 + 1..n]$ 的解

- 求解 S_3 : 跨越子数组的逆序对数目
- $S = S_1 + S_2 + S_3$

问题：如何求解 S_3 ?

合并问题解

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

?	?	?	?	?	?
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

$\{(13, 12), (15, 12), (18, 12)\}$

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	?	?	?	?	?
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

\emptyset

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	0	?	?	?	?
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

$\{(13, 9), (10, 9), (15, 9), (18, 9)\}$

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	0	4	?	?	?
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

$\{(15, 14), (18, 14)\}$

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	0	4	2	?	?
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

$\{(18, 17)\}$

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	0	4	2	1	?
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

\emptyset

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	0	4	2	1	0
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

逆序对总数：

$$3 + 4 + 2 + 1 = 10$$

3	0	4	2	1	0
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略一：直接求解

- 对每个 $A[j] \in A[m + 1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目
- 求解 S_3 的算法运行时间： $O(n^2)$

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

逆序对总数：

$$3 + 4 + 2 + 1 = 10$$

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	0	4	2	1	0
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3

- 策略一：直接求解

- 对每个 $A[j] \in A[m+1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目
- 求解 S_3 的算法运行时间： $O(n^2)$
- 分而治之框架的算法运行时间： $T(n) = 2 \cdot T(n/2) + O(n^2) \longrightarrow O(n^2)$

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

逆序对总数：

$$3 + 4 + 2 + 1 = 10$$

数组 $A[m+1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	0	4	2	1	0
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3

● 策略一：直接求解

- 对每个 $A[j] \in A[m+1..n]$ ，枚举 $A[i] \in A[1..m]$ 并统计逆序对数目

- 求解 S_3 的算法运行时间： $O(n^2)$

- 分而治之框架的算法运行时间： $T(n) = 2 \cdot T(n/2) + O(n^2) \longrightarrow O(n^2)$

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

逆序对总数：

$$3 + 4 + 2 + 1 = 10$$

数组 $A[m+1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	0	4	2	1	0
---	---	---	---	---	---

逆序对数

直接求解 S_3 的分而治之较蛮力枚举并未提高算法运行时间！

!!!

合并问题解：求解 S_3



- 致因分析

- 运行时间受制于跨越子数组的逆序对计数方法

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

$\{(13, 12), (15, 12), (18, 12)\}$

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	?	?	?	?	?
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 致因分析

- 运行时间受制于跨越子数组的逆序对计数方法
- 数组的有序性通常有助于提高算法的运行时间

数组 $A[1..m]$

6	8	10	13	15	18
---	---	----	----	----	----

先排序再二分查找。

$\{(13, 12), (15, 12), (18, 12)\}$

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

3	?	?	?	?	?
---	---	---	---	---	---

逆序对数

合并问题解：求解 S_3



- 策略二：排序求解

数组 $A[1..m]$

13	8	10	6	15	18
----	---	----	---	----	----

数组 $A[m + 1..n]$

12	20	9	14	17	19
----	----	---	----	----	----

逆序对数：

?	?	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略二：排序求解
 - 分别对数组 $A[1..m]$ 和 $A[m + 1..n]$ 进行排序

数组 $A[1..m]$

6	8	10	13	15	18
---	---	----	----	----	----

数组 $A[m + 1..n]$

9	12	14	17	19	20
---	----	----	----	----	----

逆序对数：

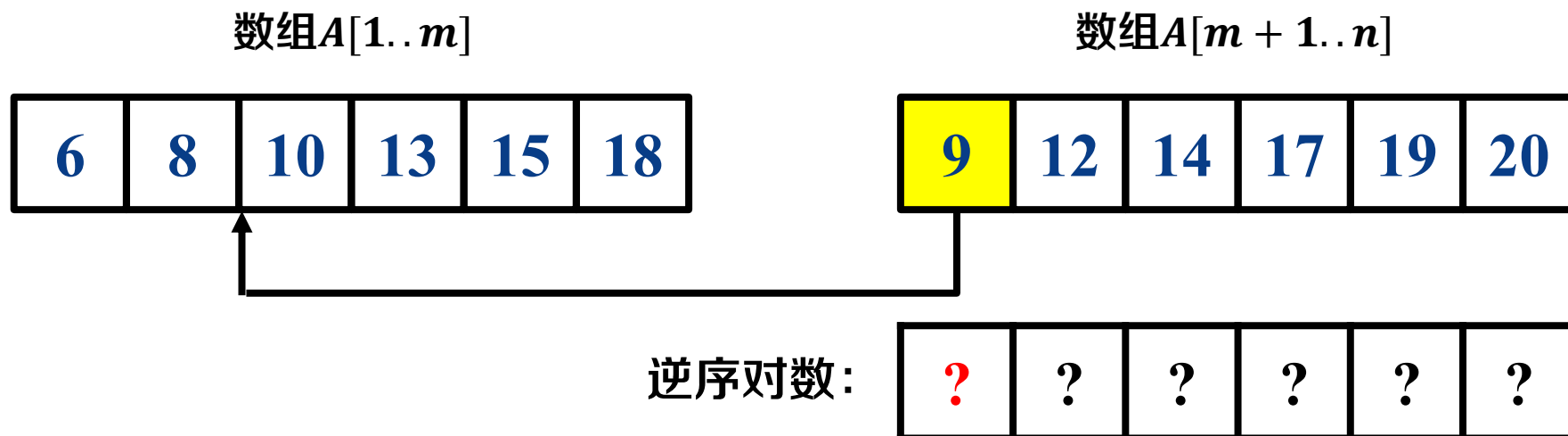
?	?	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略二：排序求解

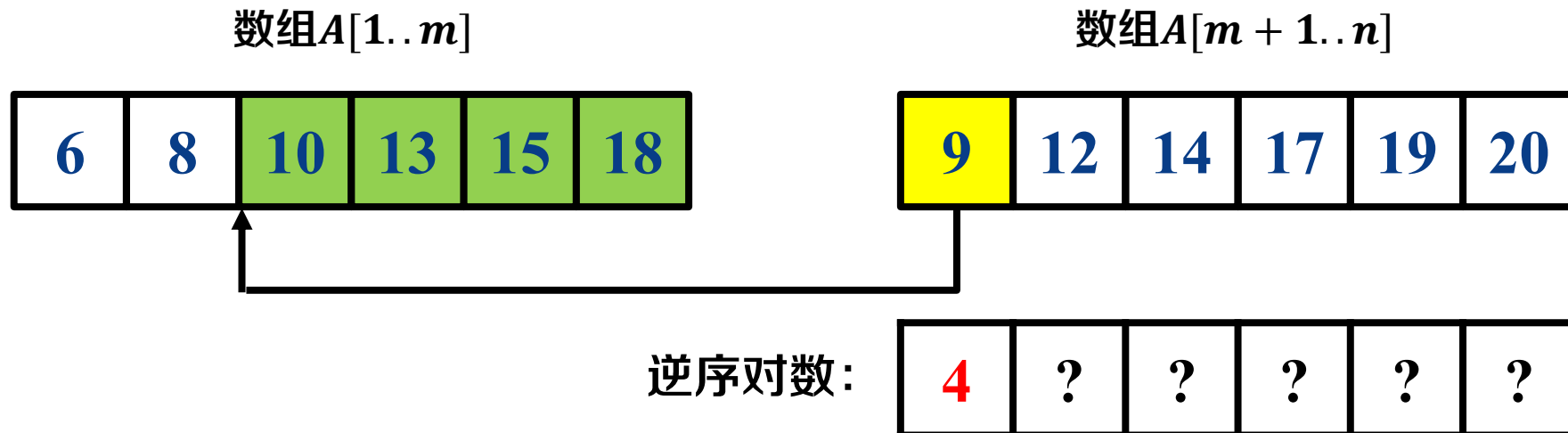
- 分别对数组 $A[1..m]$ 和 $A[m + 1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m + 1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位



合并问题解：求解 S_3

● 策略二：排序求解

- 分别对数组 $A[1..m]$ 和 $A[m + 1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m + 1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位
- $A[j]$ 在 $A[1..m]$ 定位点右侧的元素均可与 $A[j]$ 构成逆序对



合并问题解：求解 S_3



- 策略二：排序求解

- 分别对数组 $A[1..m]$ 和 $A[m+1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m+1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位
- $A[j]$ 在 $A[1..m]$ 定位点右侧的元素均可与 $A[j]$ 构成逆序对



合并问题解：求解 S_3



- 策略二：排序求解

- 分别对数组 $A[1..m]$ 和 $A[m + 1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m + 1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位
- $A[j]$ 在 $A[1..m]$ 定位点右侧的元素均可与 $A[j]$ 构成逆序对

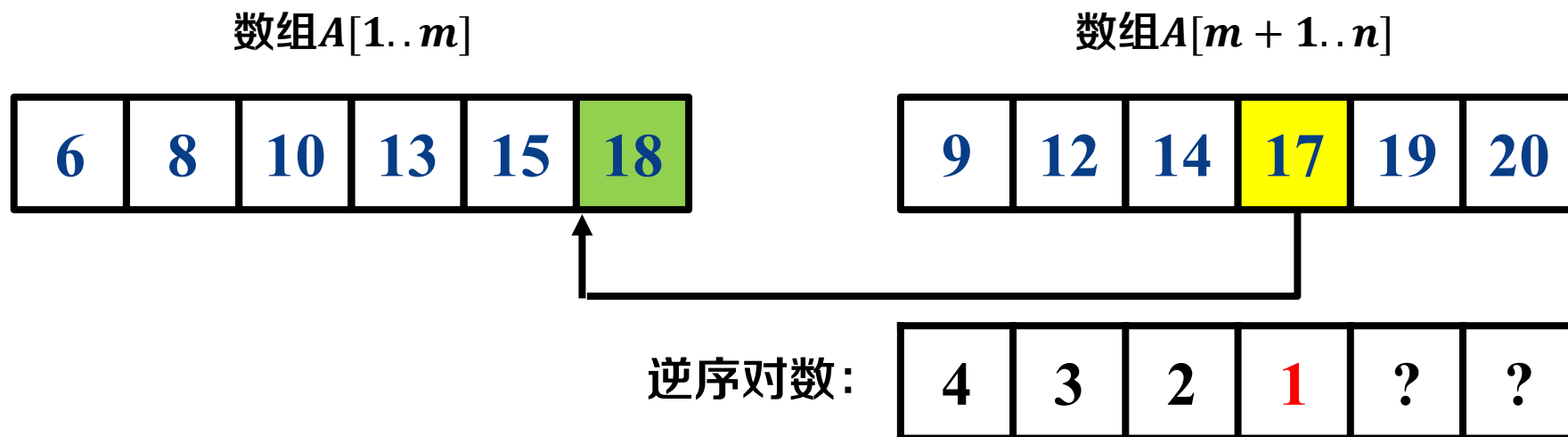


合并问题解：求解 S_3



- 策略二：排序求解

- 分别对数组 $A[1..m]$ 和 $A[m+1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m+1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位
- $A[j]$ 在 $A[1..m]$ 定位点右侧的元素均可与 $A[j]$ 构成逆序对

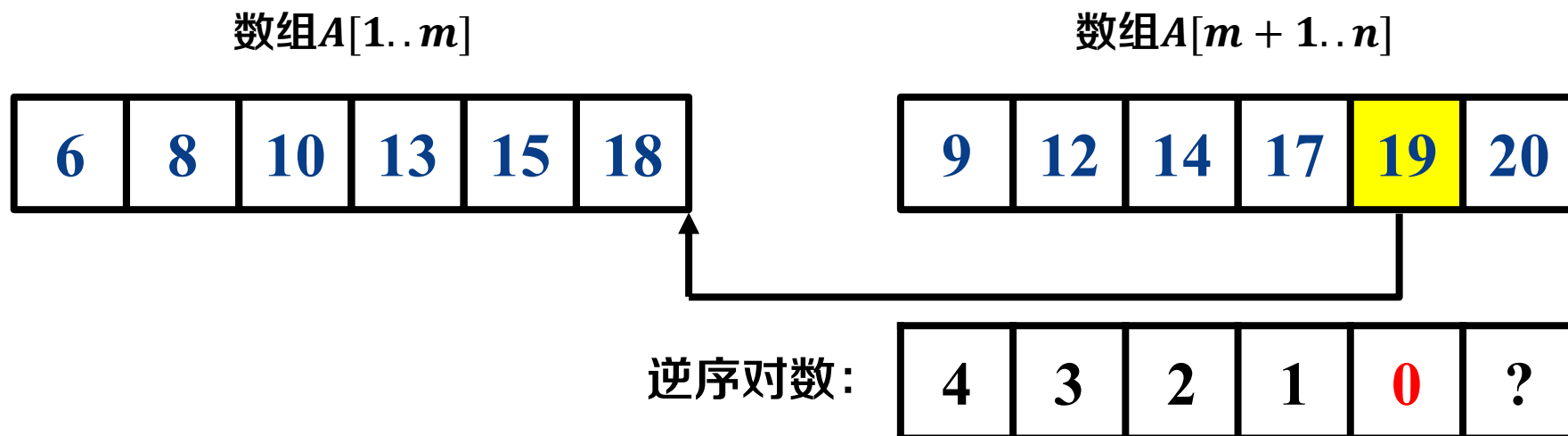


合并问题解：求解 S_3



- 策略二：排序求解

- 分别对数组 $A[1..m]$ 和 $A[m+1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m+1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位
- $A[j]$ 在 $A[1..m]$ 定位点右侧的元素均可与 $A[j]$ 构成逆序对

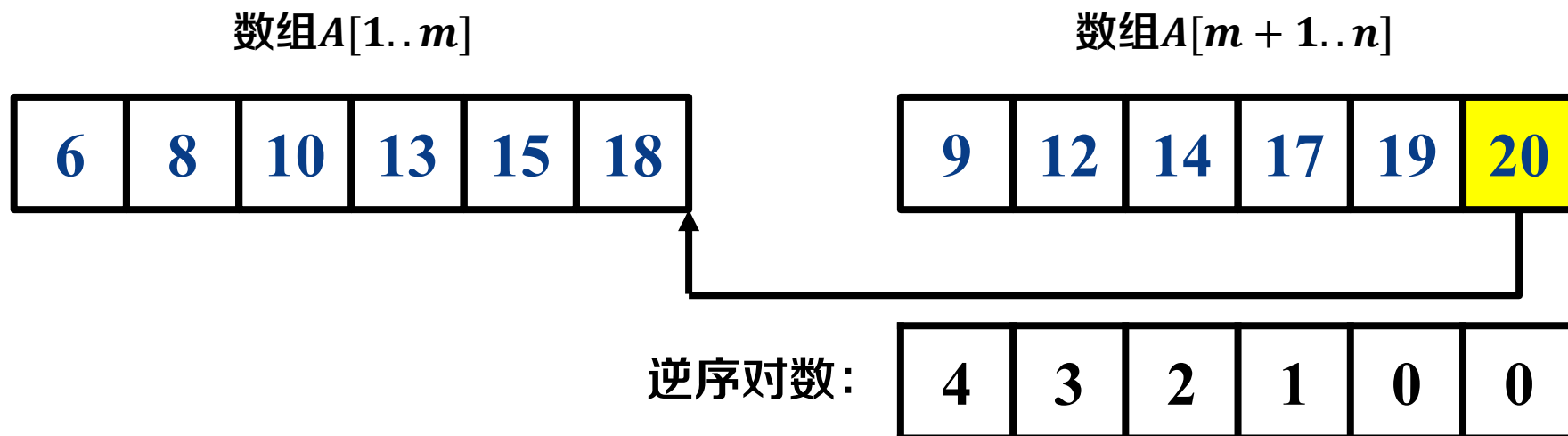


合并问题解：求解 S_3



- 策略二：排序求解

- 分别对数组 $A[1..m]$ 和 $A[m+1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m+1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位
- $A[j]$ 在 $A[1..m]$ 定位点右侧的元素均可与 $A[j]$ 构成逆序对



合并问题解：求解 S_3



- 策略二：排序求解

- 分别对数组 $A[1..m]$ 和 $A[m + 1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m + 1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位
- **$A[j]$ 在 $A[1..m]$ 定位点右侧的元素**均可与 $A[j]$ 构成逆序对
- 求解 S_3 的算法运行时间： $O(n \log n)$

$$\left. \begin{array}{l} \text{分别对数组 } A[1..m] \text{ 和 } A[m+1..n] \text{ 进行排序} \\ \text{对于每个 } A[j] \in A[m+1..n], \text{ 采用二分查找为其在 } A[1..m] \text{ 中定位} \end{array} \right\} \frac{n}{2} \log \frac{n}{2}$$

合并问题解：求解 S_3

● 策略二：排序求解

- 分别对数组 $A[1..m]$ 和 $A[m+1..n]$ 进行**排序**
- 对于每个 $A[j] \in A[m+1..n]$ ，采用**二分查找**为其在 $A[1..m]$ 中定位
- $A[j]$ 在 $A[1..m]$ 定位点右侧的元素均可与 $A[j]$ 构成逆序对
- 求解 S_3 的算法运行时间： $O(n \log n)$
- 分治框架的算法运行时间： $T(n) = 2T(n/2) + O(n \log n) \longrightarrow O(n \log^2 n)$

扩展的主定理求解
比 $O(n^2)$ 优

排序求解 S_3 的分而治之提高了算法运行时间，是否还有优化可能？

合并问题解：求解 S_3



- 算法优化
 - 排序和二分查找均无再优化空间

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

合并问题解：求解 S_3

- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

13	8	10
----	---	----

6	15	18
---	----	----

12	20	9
----	----	---

14	17	19
----	----	----

合并问题解：求解 S_3

- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

8	10	13
---	----	----

排序

6	15	18
---	----	----

排序

12	20	9
----	----	---

14	17	19
----	----	----

合并问题解：求解 S_3

- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

8	10	13
---	----	----

6	15	18
---	----	----

12	20	9
----	----	---

14	17	19
----	----	----

二分查找

合并问题解：求解 S_3



- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

8	10	13
---	----	----

6	15	18
---	----	----

9	12	20
---	----	----

排序

14	17	19
----	----	----

排序

合并问题解：求解 S_3

- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

8	10	13
---	----	----

6	15	18
---	----	----

9	12	20
---	----	----

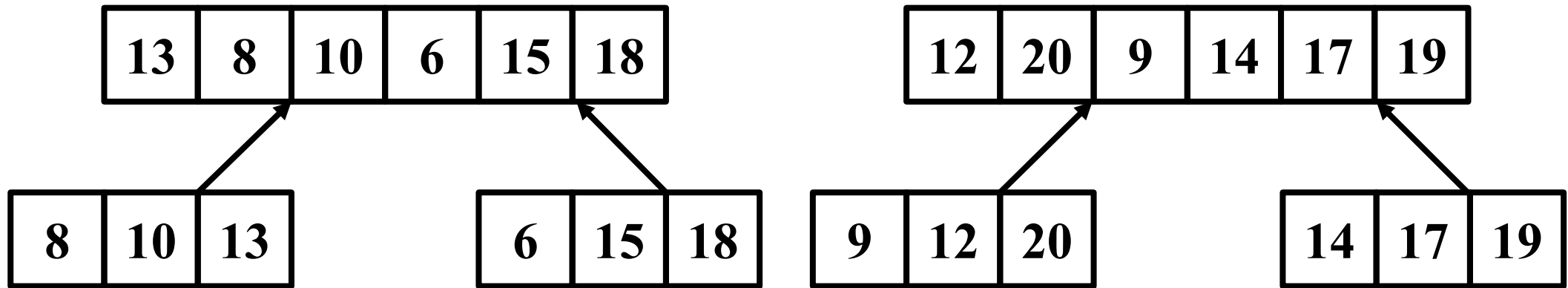
14	17	19
----	----	----

二分查找

合并问题解：求解 S_3

- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

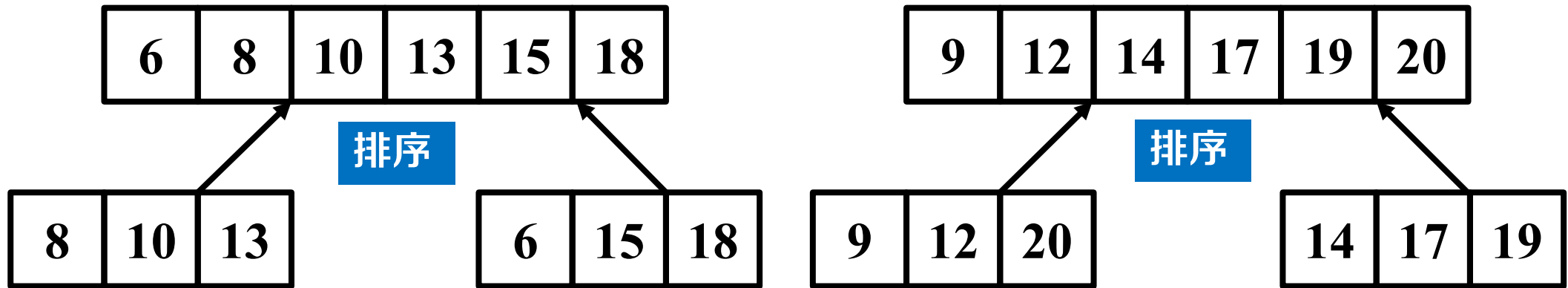
子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$



合并问题解：求解 S_3

- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

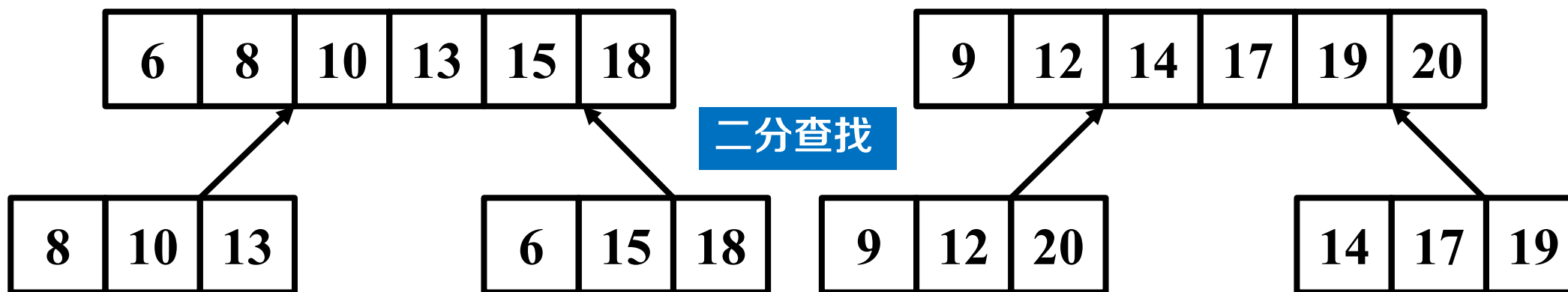


合并问题解：求解 S_3



- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

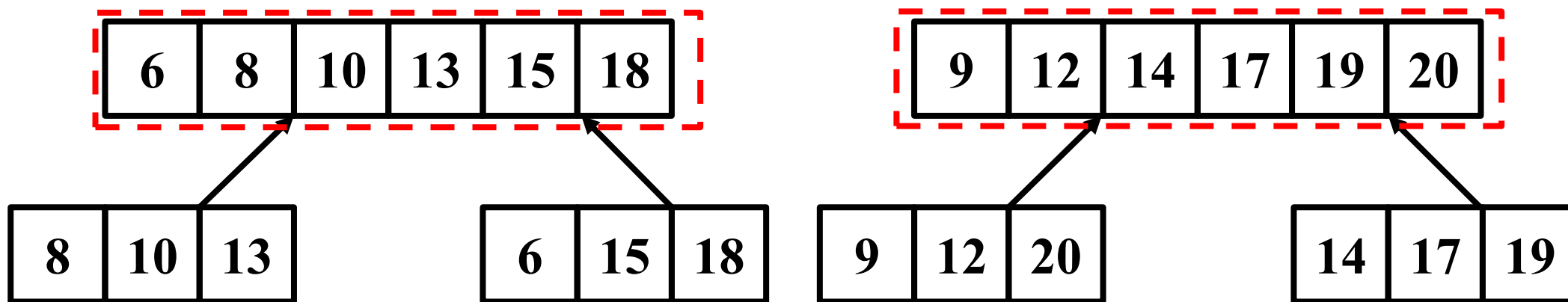


合并问题解：求解 S_3

- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

子数组排序的运行时间： $O(n \log n)$
二分查找的运行时间： $O(n \log n)$

排序未利用子
数组有序性质！



问题：如何将排序过程融入算法框架？

合并问题解：求解 S_3

- 算法优化
 - 排序和二分查找均无再优化空间
- 致因分析
 - 未将排序过程融入整个算法框架

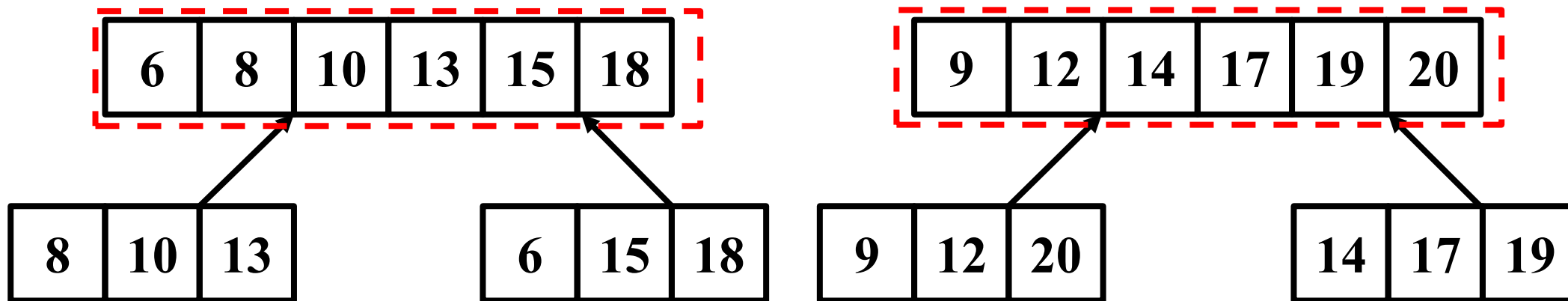
子数组排序的运行时间： $O(n \log n)$

二分查找的运行时间： $O(n \log n)$

可用归并排序优化父数组生成策略

可将时间优化到 $O(n)$

排序未利用子数组有序性质！



问题：如何将排序过程融入算法框架？归并排序！

合并问题解：求解 S_3



- 算法优化
 - 合并问题解的同时对数组进行排序

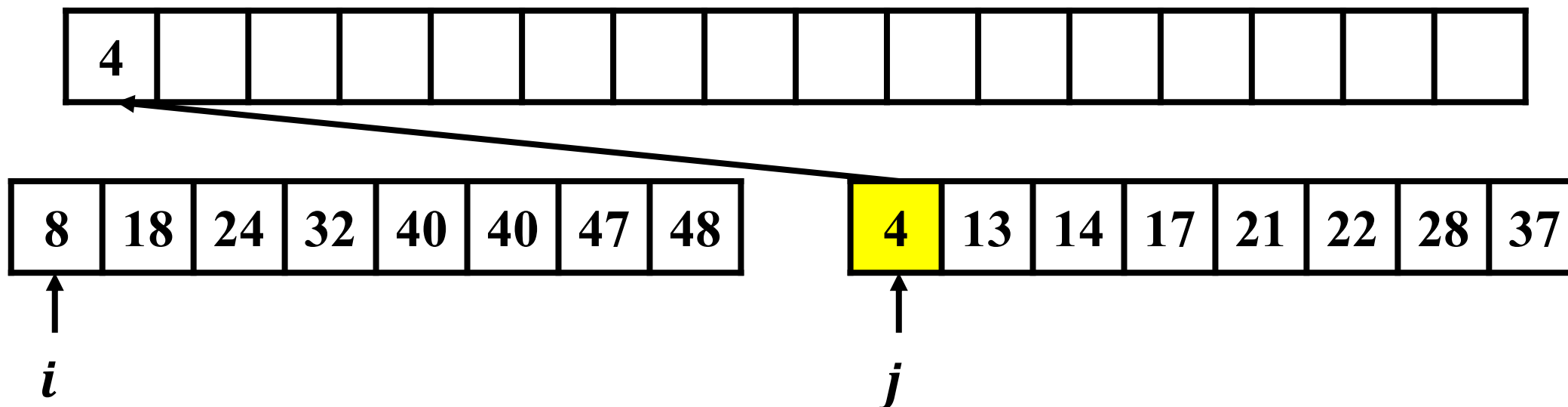
子数组排序的运行时间： $O(n)$
二分查找的运行时间： ?

合并问题解：求解 S_3



- 算法优化
 - 合并问题解的同时对数组进行排序
- 规律发现
 - 归并过程中可同时计算逆序对数目

子数组排序的运行时间： $O(n)$
二分查找的运行时间： ?



合并问题解：求解 S_3

● 算法优化

- 合并问题解的同时对数组进行排序

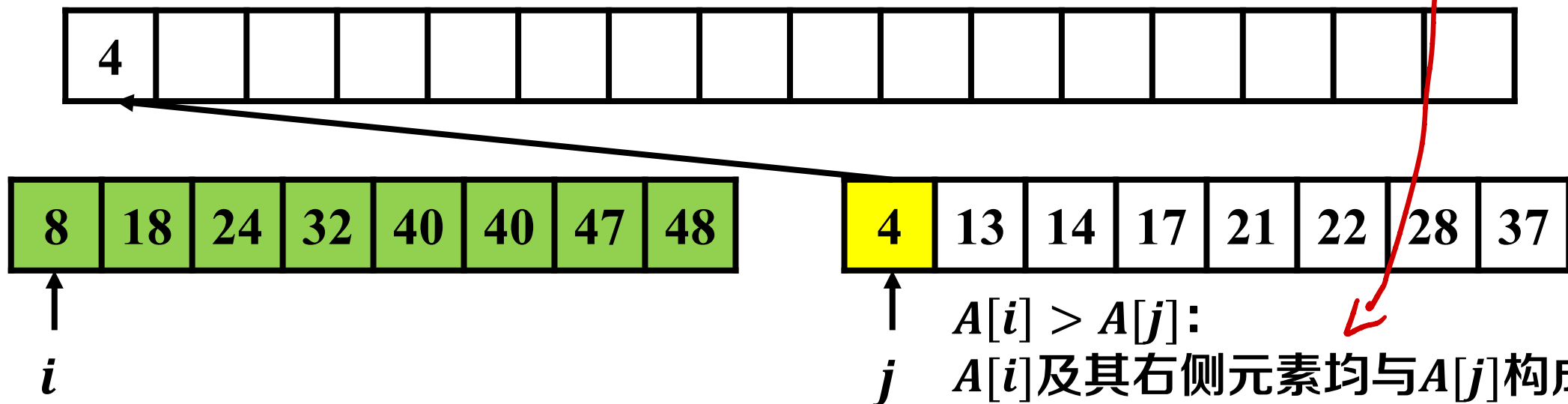
子数组排序的运行时间： $O(n)$

二分查找的运行时间：？

● 规律发现

- 归并过程中可同时计算逆序对数目

归并过程中无形中解决逆序对计数问题

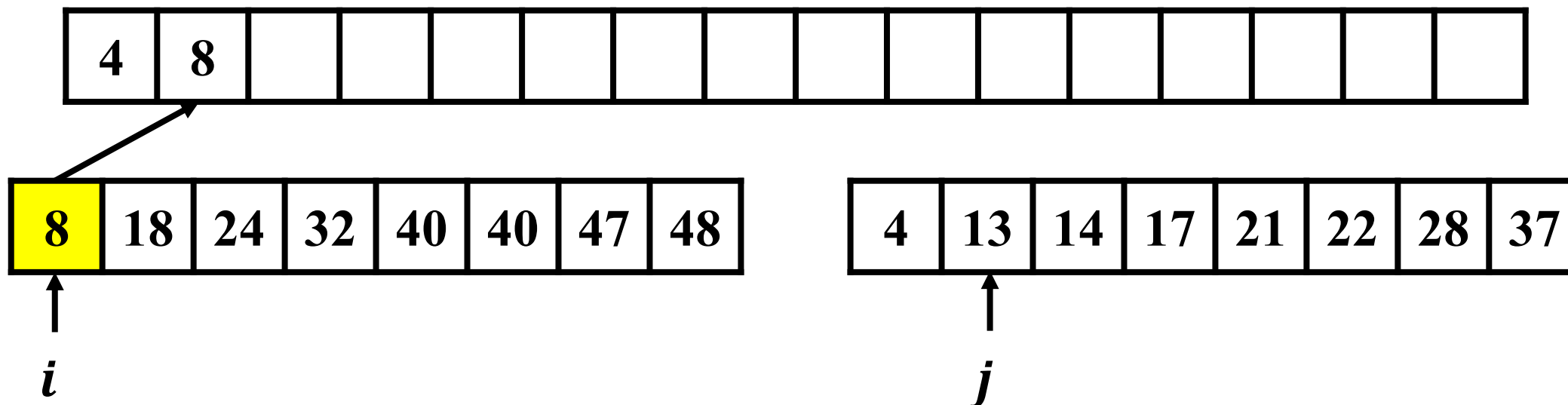


合并问题解：求解 S_3



- 算法优化
 - 合并问题解的同时对数组进行排序
- 规律发现
 - 归并过程中可同时计算逆序对数目

子数组排序的运行时间： $O(n)$
二分查找的运行时间：？

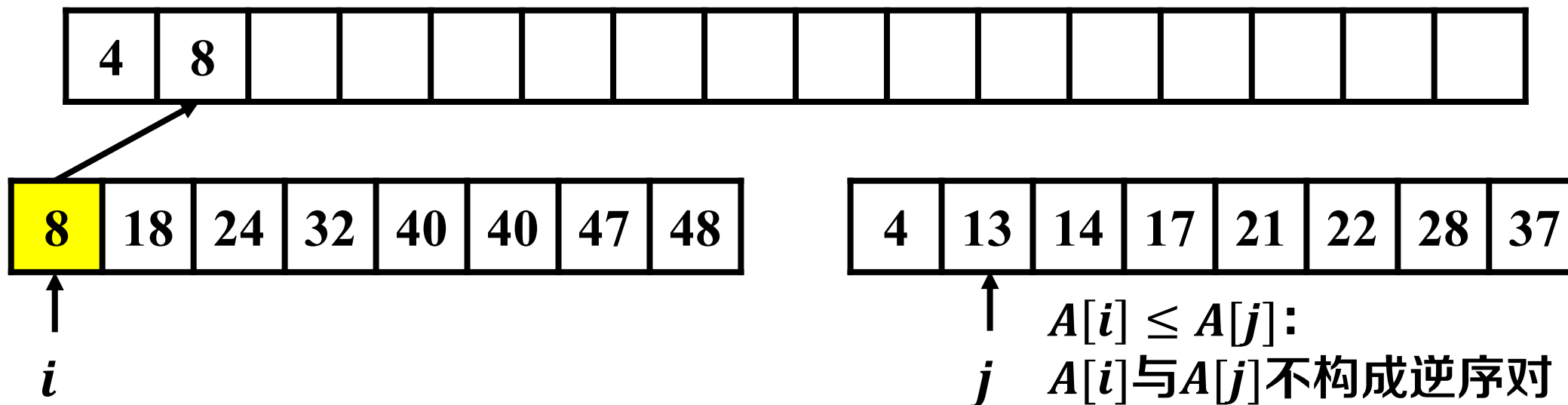


合并问题解：求解 S_3



- 算法优化
 - 合并问题解的同时对数组进行排序
- 规律发现
 - 归并过程中可同时计算逆序对数目

子数组排序的运行时间： $O(n)$
二分查找的运行时间：？

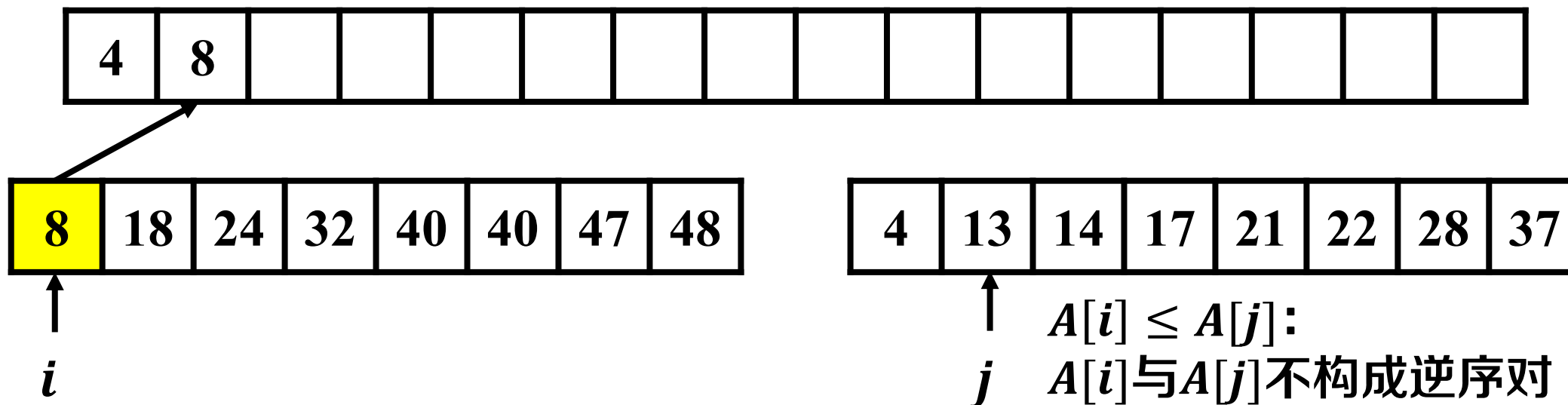


合并问题解：求解 S_3



- 算法优化
 - 合并问题解的同时对数组进行排序
- 规律发现
 - 归并过程中可同时计算逆序对数目

子数组排序的运行时间： $O(n)$
二分查找的运行时间：？



合并问题解：求解 S_3



- 策略三：归并求解

- 从左到右扫描有序子数组： $A[i] \in A[1..m]$, $A[j] \in A[m+1..n]$

☆ {
 ○ 如果 $A[i] > A[j]$, 统计逆序对, j 向右移
 ○ 如果 $A[i] \leq A[j]$, i 向右移

☆ 精髓

- 利用归并排序框架保证合并后数组的有序性

合并问题解：求解 S_3



- 策略三：归并求解



子数组:

6	8	10	13	15	18
---	---	----	----	----	----



$i = 1$

9	12	14	17	19	20
---	----	----	----	----	----



$j = 7$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移

若 $A[i] \leq A[j]$, i 向右移

逆序对数:

?	?	?	?	?	?
---	---	---	---	---	---

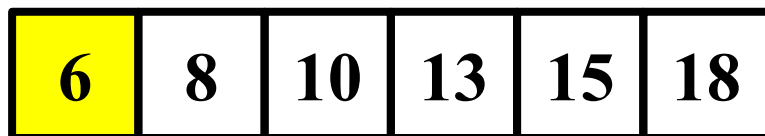
合并问题解：求解 S_3



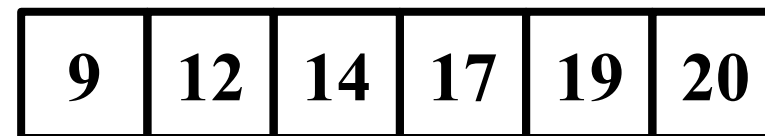
- 策略三：归并求解



子数组:



↑
 $i = 1$

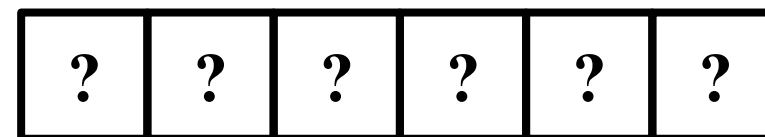


↑
 $j = 7$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:



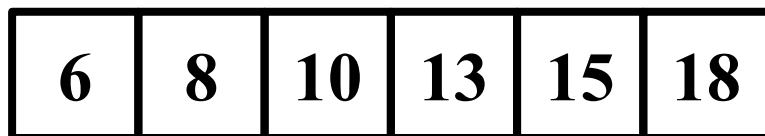
合并问题解：求解 S_3



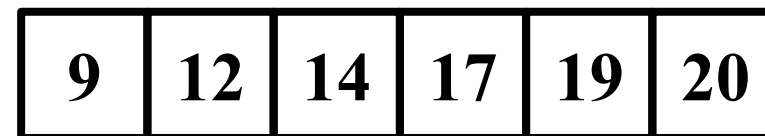
- 策略三：归并求解



子数组:



↑
 $i = 2$

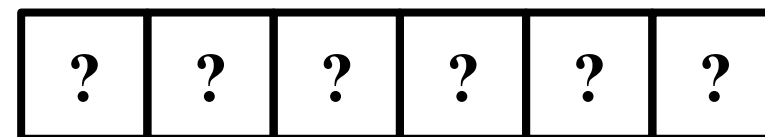


↑
 $j = 7$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移
若 $A[i] \leq A[j]$, i 向右移

逆序对数:



合并问题解：求解 S_3



- 策略三：归并求解

6	8										
---	---	--	--	--	--	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----



$i = 2$

9	12	14	17	19	20
---	----	----	----	----	----



$j = 7$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移

若 $A[i] \leq A[j]$, i 向右移

逆序对数:

?	?	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8										
---	---	--	--	--	--	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

↑
 $i = 3$

9	12	14	17	19	20
---	----	----	----	----	----

↑
 $j = 7$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

?	?	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9									
---	---	---	--	--	--	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----



$i = 3$

9	12	14	17	19	20
---	----	----	----	----	----



$j = 7$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移

若 $A[i] \leq A[j]$, i 向右移

逆序对数:

4	?	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9									
---	---	---	--	--	--	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----



$i = 3$

9	12	14	17	19	20
---	----	----	----	----	----



$j = 8$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	?	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10								
---	---	---	----	--	--	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----



$i = 3$

9	12	14	17	19	20
---	----	----	----	----	----



$j = 8$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移

若 $A[i] \leq A[j]$, i 向右移

逆序对数:

4	?	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10								
---	---	---	----	--	--	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 4$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 8$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	?	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12							
---	---	---	----	----	--	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 4$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 8$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12							
---	---	---	----	----	--	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

↑
 $i = 4$

9	12	14	17	19	20
---	----	----	----	----	----

↑
 $j = 9$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移
若 $A[i] \leq A[j]$, i 向右移

逆序对数:

4	3	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13						
---	---	---	----	----	----	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 4$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 9$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13						
---	---	---	----	----	----	--	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 5$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 9$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移

若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	?	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14					
---	---	---	----	----	----	----	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 5$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 9$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	2	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14					
---	---	---	----	----	----	----	--	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 5$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 10$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	2	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14	15				
---	---	---	----	----	----	----	----	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 5$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 10$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	2	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14	15				
---	---	---	----	----	----	----	----	--	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

↑
 $i = 6$

9	12	14	17	19	20
---	----	----	----	----	----

↑
 $j = 10$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	2	?	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14	15	17			
---	---	---	----	----	----	----	----	----	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 6$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 10$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移
若 $A[i] \leq A[j]$, i 向右移

逆序对数:

4	3	2	1	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14	15	17			
---	---	---	----	----	----	----	----	----	--	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

↑
 $i = 6$

9	12	14	17	19	20
---	----	----	----	----	----

↑
 $j = 11$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移
若 $A[i] \leq A[j]$, i 向右移

逆序对数:

4	3	2	1	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14	15	17	18		
---	---	---	----	----	----	----	----	----	----	--	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----

$i = 6$

9	12	14	17	19	20
---	----	----	----	----	----

$j = 11$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	2	1	?	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14	15	17	18	19	
---	---	---	----	----	----	----	----	----	----	----	--

子数组:

6	8	10	13	15	18
---	---	----	----	----	----



9	12	14	17	19	20
---	----	----	----	----	----



$j = 11$

扫描子数组:

若 $A[i] > A[j]$, 统计逆序对, j 向右移
若 $A[i] \leq A[j]$, i 向右移

逆序对数:

4	3	2	1	0	?
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14	15	17	18	19	20
---	---	---	----	----	----	----	----	----	----	----	----

子数组:

6	8	10	13	15	18
---	---	----	----	----	----



9	12	14	17	19	20
---	----	----	----	----	----



$j = 12$

扫描子数组:

若 $A[i] > A[j]$ ，统计逆序对， j 向右移
若 $A[i] \leq A[j]$ ， i 向右移

逆序对数:

4	3	2	1	0	0
---	---	---	---	---	---

合并问题解：求解 S_3



- 策略三：归并求解

6	8	9	10	12	13	14	15	17	18	19	20
---	---	---	----	----	----	----	----	----	----	----	----

子数组:

6	8	10	13	15	18
---	---	----	----	----	----



9	12	14	17	19	20
---	----	----	----	----	----



$$S_3 = 4 + 3 + 2 + 1 = 10$$

逆序对数:

4	3	2	1	0	0
---	---	---	---	---	---

求解 S_3 时间复杂度降至 $O(n)$

- 归并求解: $\text{MergeCount}(A, \text{left}, \text{mid}, \text{right})$

输入: 数组 $A[1..n]$, 数组下标 $\text{left}, \text{mid}, \text{right}$

输出: 跨越数组 $A[\text{left}..\text{mid}]$ 和 $A[\text{mid} + 1..\text{right}]$ 的逆序对数, $A[\text{left}..\text{right}]$

$A'[\text{left}..\text{right}] \leftarrow A[\text{left}..\text{right}], S_3 \leftarrow 0$

$i \leftarrow \text{left}, j \leftarrow \text{mid} + 1, k \leftarrow 0$

while $i \leq \text{mid}$ **and** $j \leq \text{right}$ **do**

if $A'[i] \leq A'[j]$ **then**

$A[\text{left} + k] \leftarrow A'[i]$

$k \leftarrow k + 1, i \leftarrow i + 1$

end

else

$A[\text{left} + k] \leftarrow A'[j]$

$S_3 \leftarrow S_3 + (\text{mid} - i + 1)$

$k \leftarrow k + 1, j \leftarrow j + 1$

end

end

if $i \leq \text{mid}$ **then**

$A[k..\text{right}] \leftarrow A'[i..\text{mid}]$

end

else

$A[k..\text{right}] \leftarrow A'[j..\text{right}]$

end

return $S_3, A[\text{left}..\text{right}]$

遍历子数组时计算 S_3

去掉这行, 就是
归并排序

- 归并求解: $\text{MergeCount}(A, \text{left}, \text{mid}, \text{right})$

输入: 数组 $A[1..n]$, 数组下标 $\text{left}, \text{mid}, \text{right}$

输出: 跨越数组 $A[\text{left}..\text{mid}]$ 和 $A[\text{mid} + 1..\text{right}]$ 的逆序对数, $A[\text{left}..\text{right}]$

$A'[\text{left}..\text{right}] \leftarrow A[\text{left}..\text{right}], S_3 \leftarrow 0$

$i \leftarrow \text{left}, j \leftarrow \text{mid} + 1, k \leftarrow 0$

while $i \leq \text{mid}$ **and** $j \leq \text{right}$ **do**

if $A'[i] \leq A'[j]$ **then**

$A[\text{left} + k] \leftarrow A'[i]$

$k \leftarrow k + 1, i \leftarrow i + 1$

end

else

$A[\text{left} + k] \leftarrow A'[j]$

$S_3 \leftarrow S_3 + (\text{mid} - i + 1)$

$k \leftarrow k + 1, j \leftarrow j + 1$

end

end

if $i \leq \text{mid}$ **then**

$A[k..\text{right}] \leftarrow A'[i..\text{mid}]$

end

else

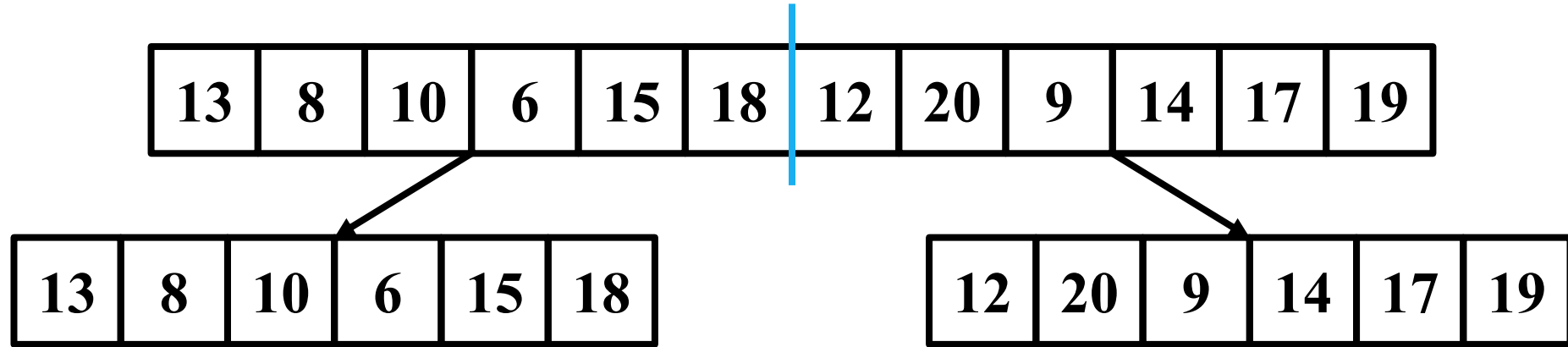
$A[k..\text{right}] \leftarrow A'[j..\text{right}]$

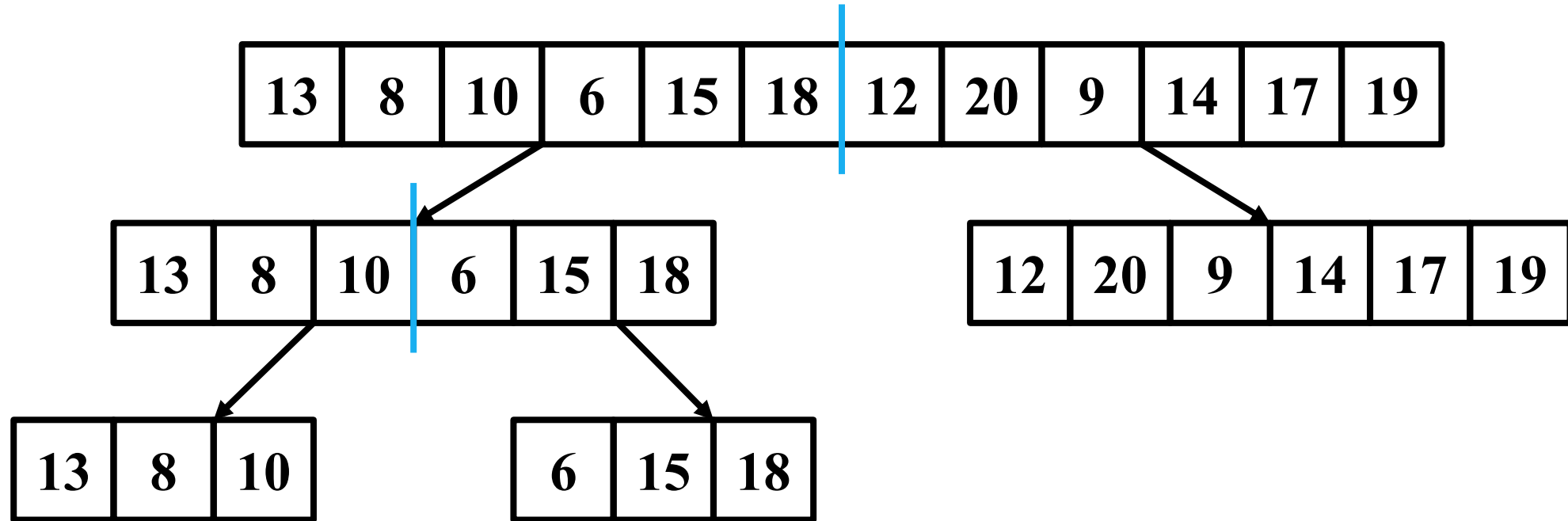
end

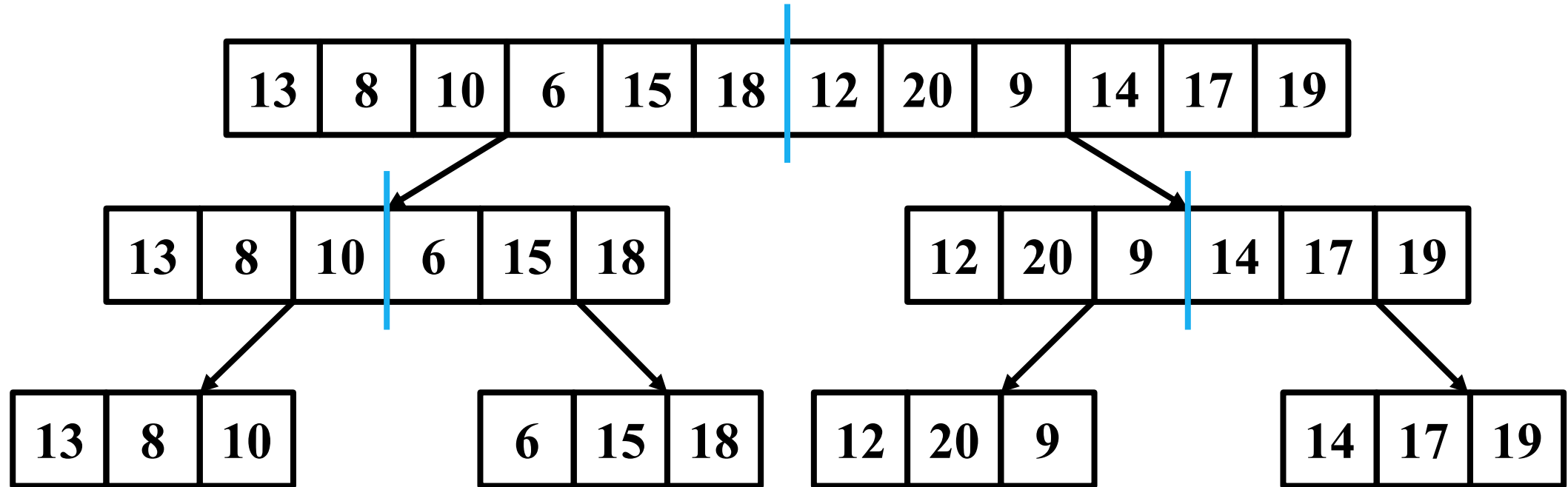
return $S_3, A[\text{left}..\text{right}]$

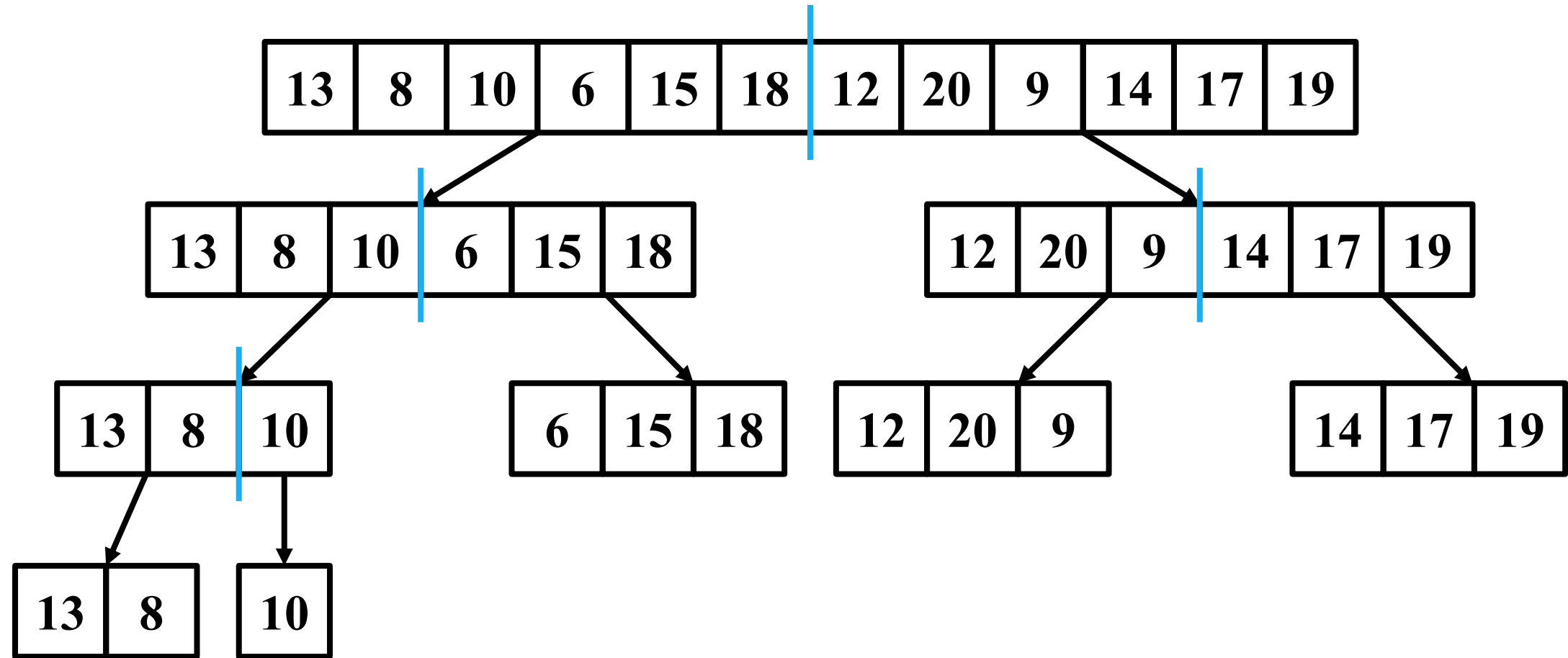
添加剩余元素保证有序

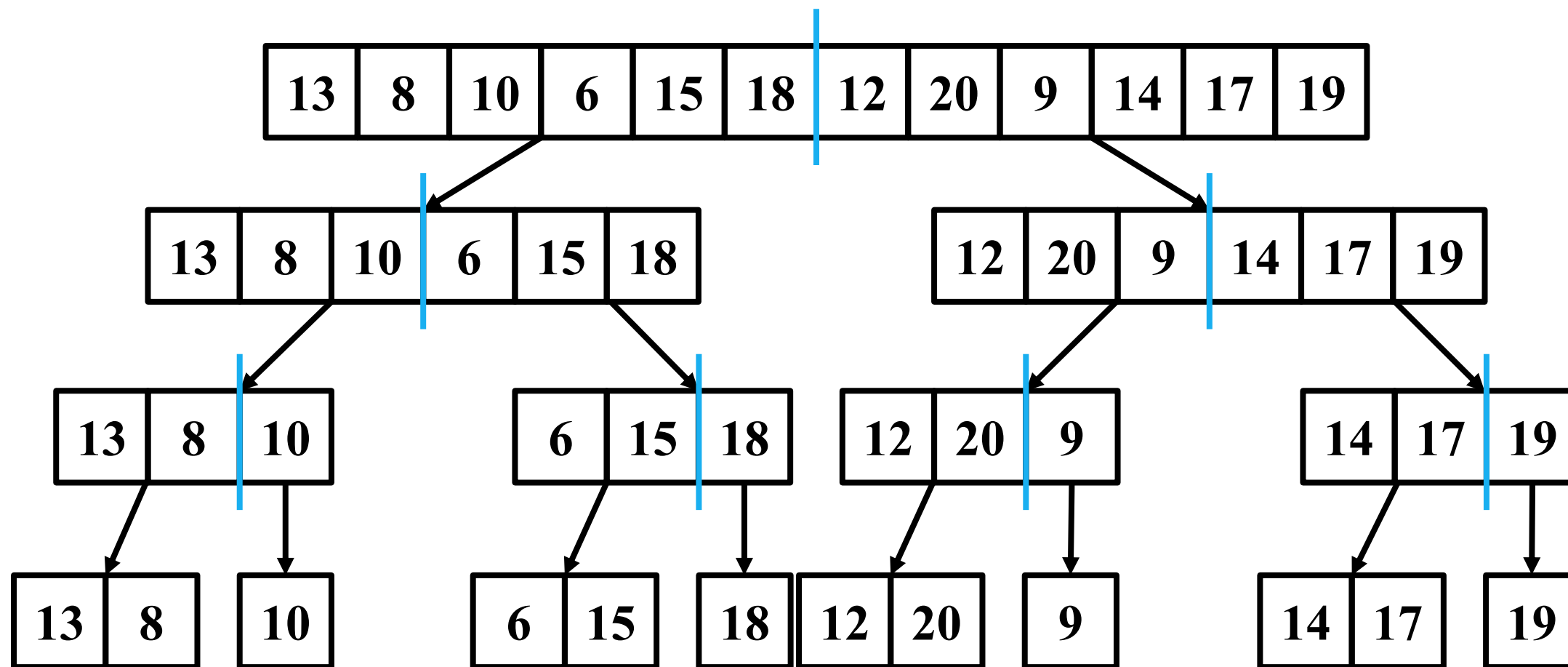
13	8	10	6	15	18	12	20	9	14	17	19
----	---	----	---	----	----	----	----	---	----	----	----

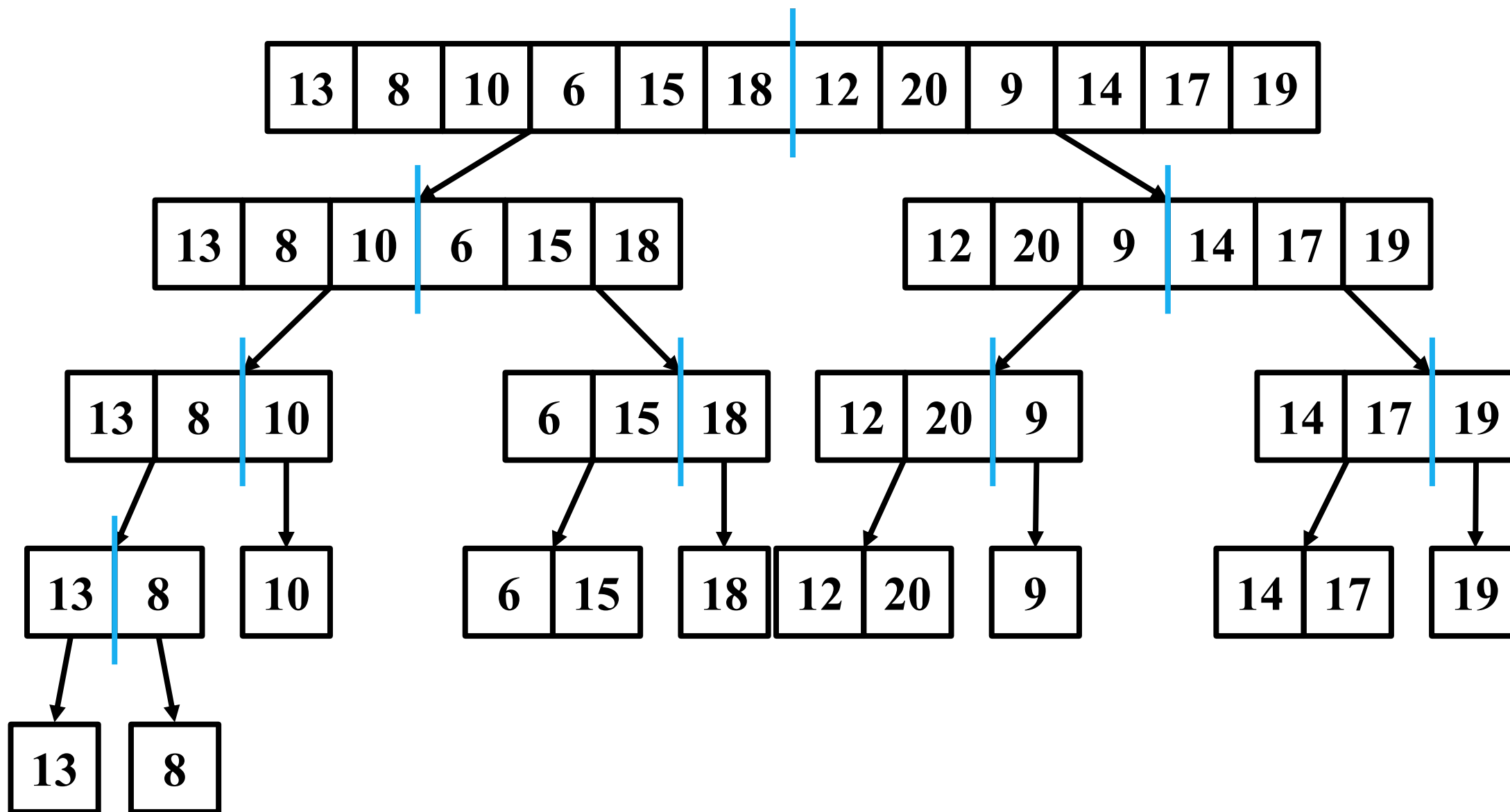


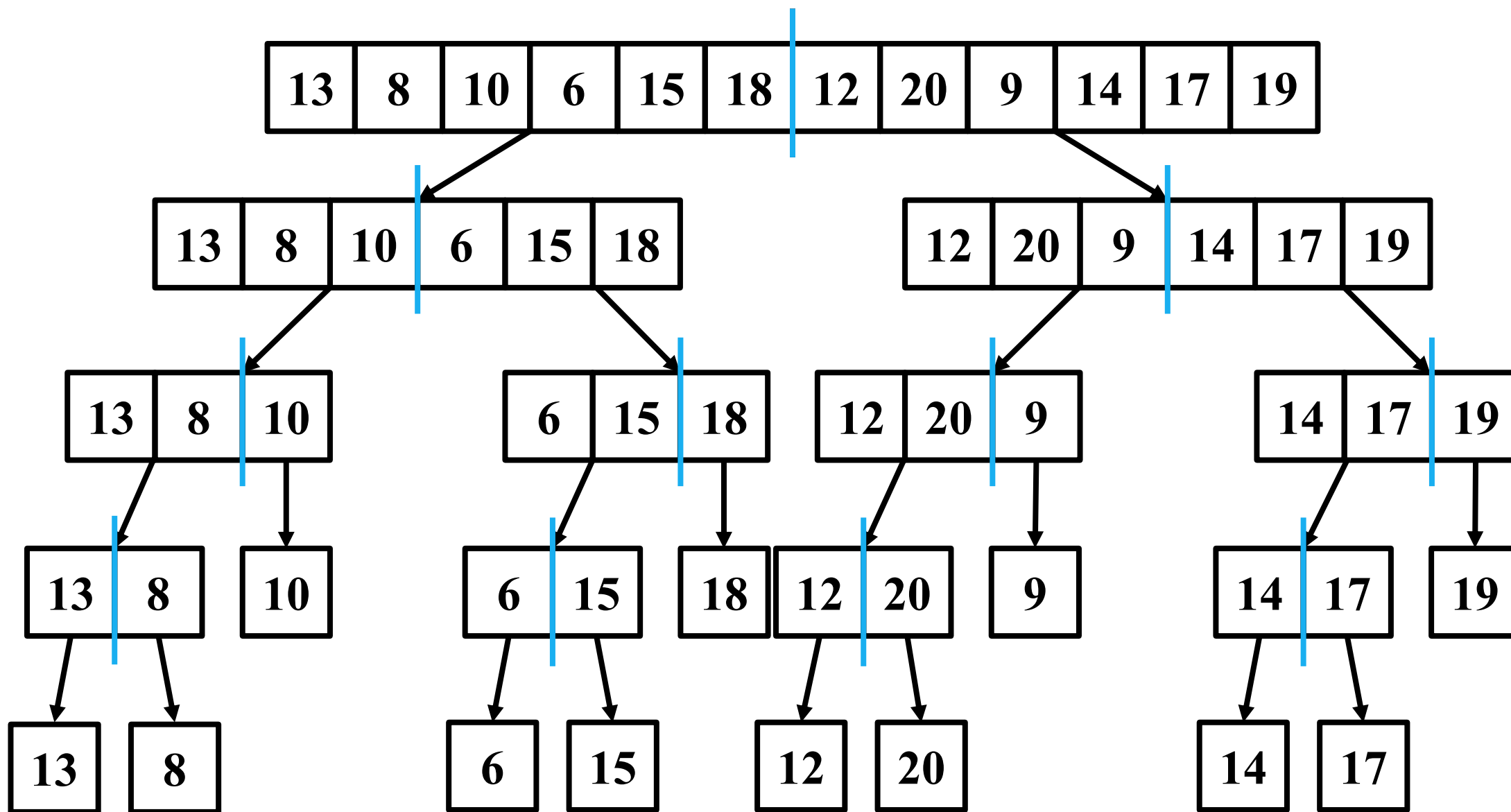




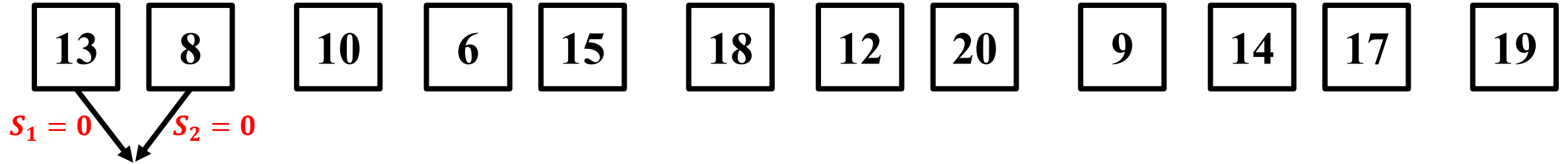


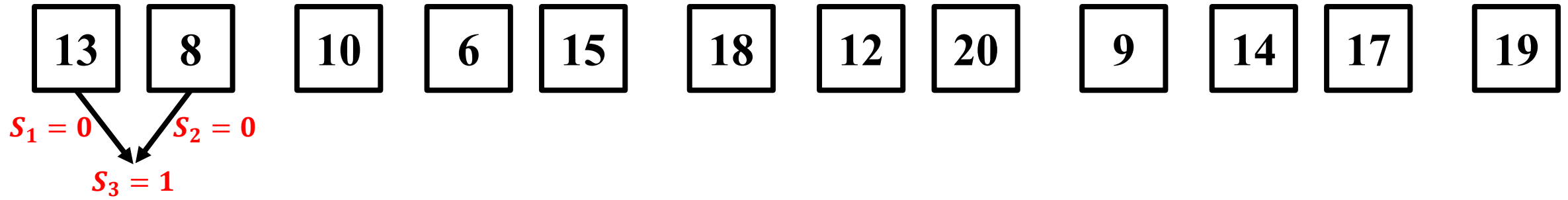


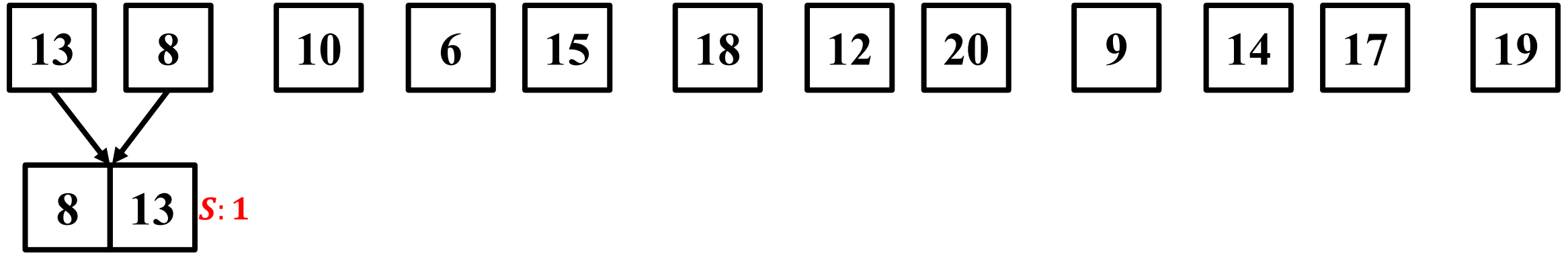


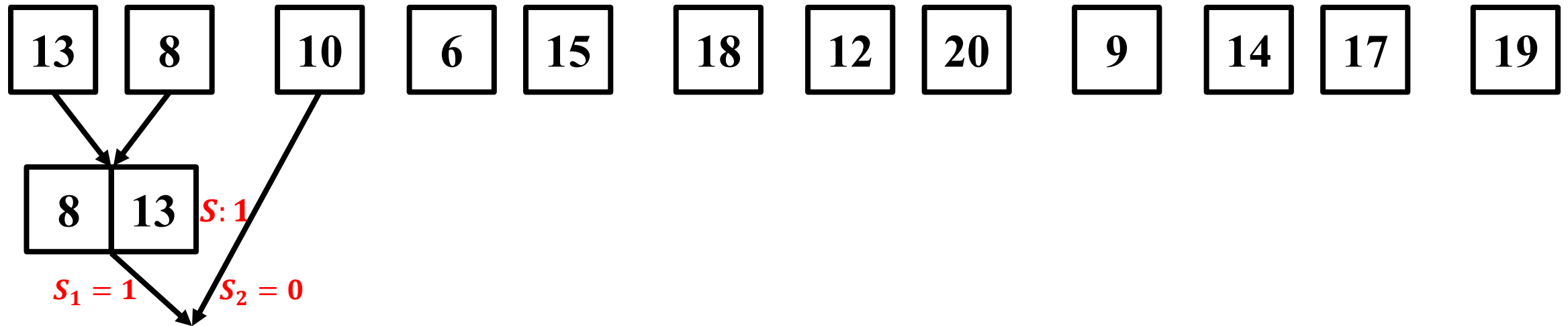


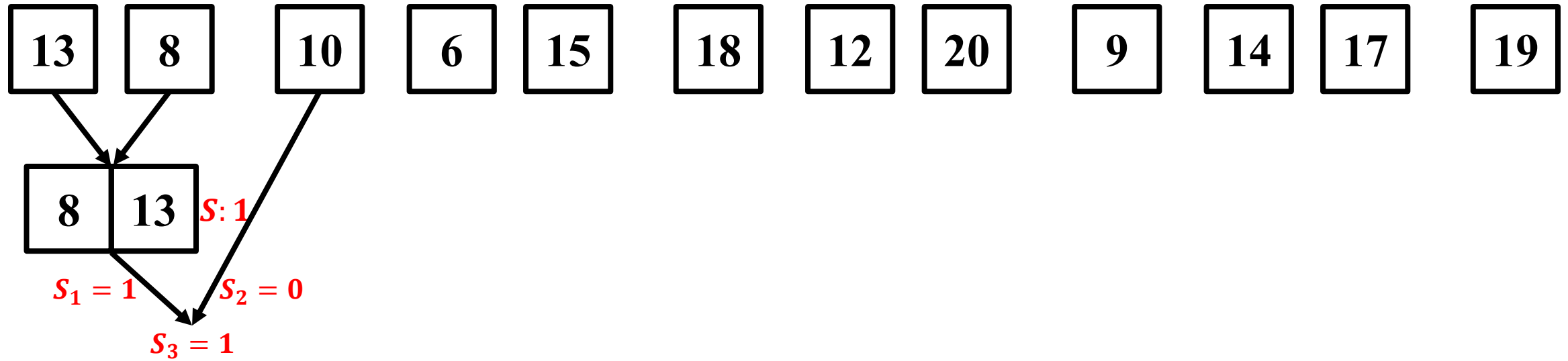
13	8	10	6	15	18	12	20	9	14	17	19
----	---	----	---	----	----	----	----	---	----	----	----

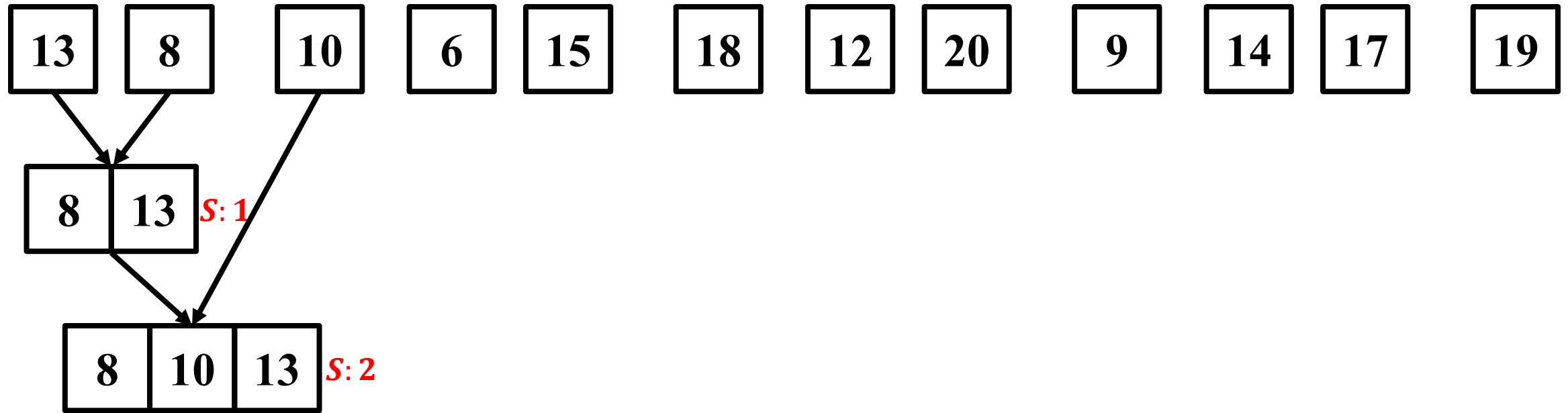


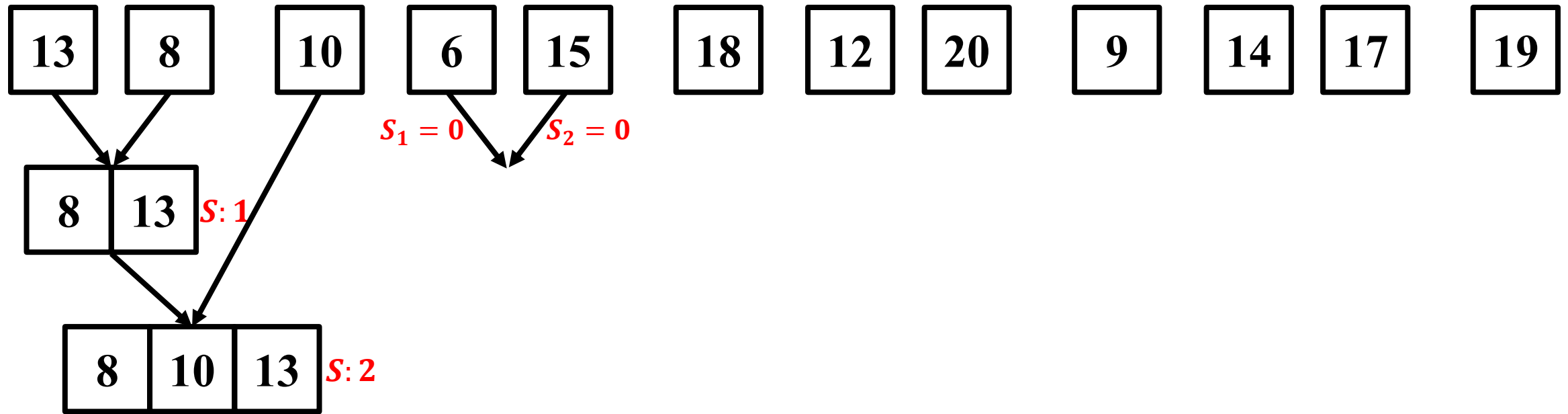


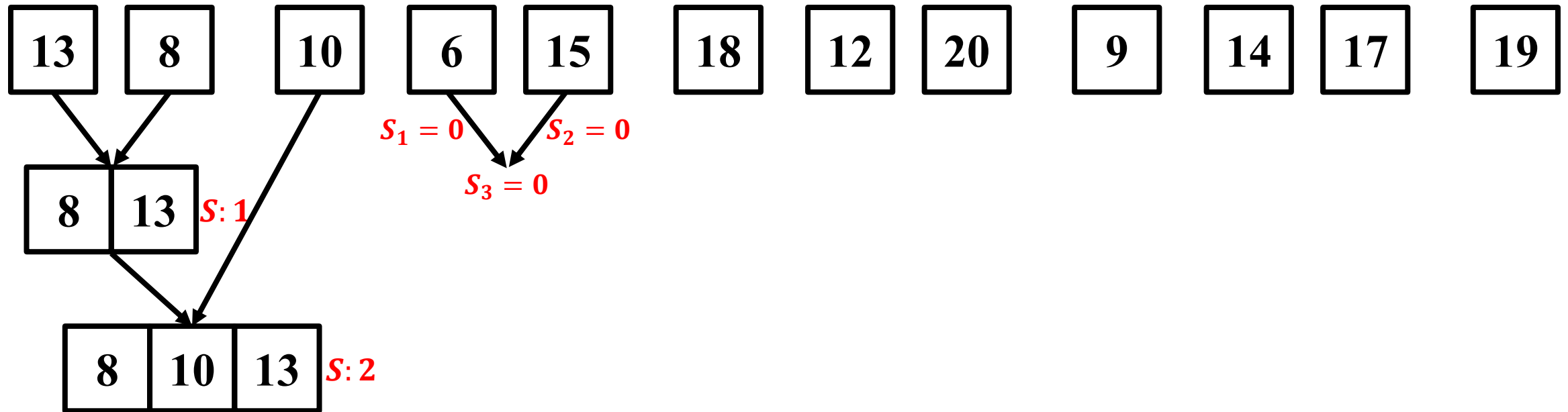


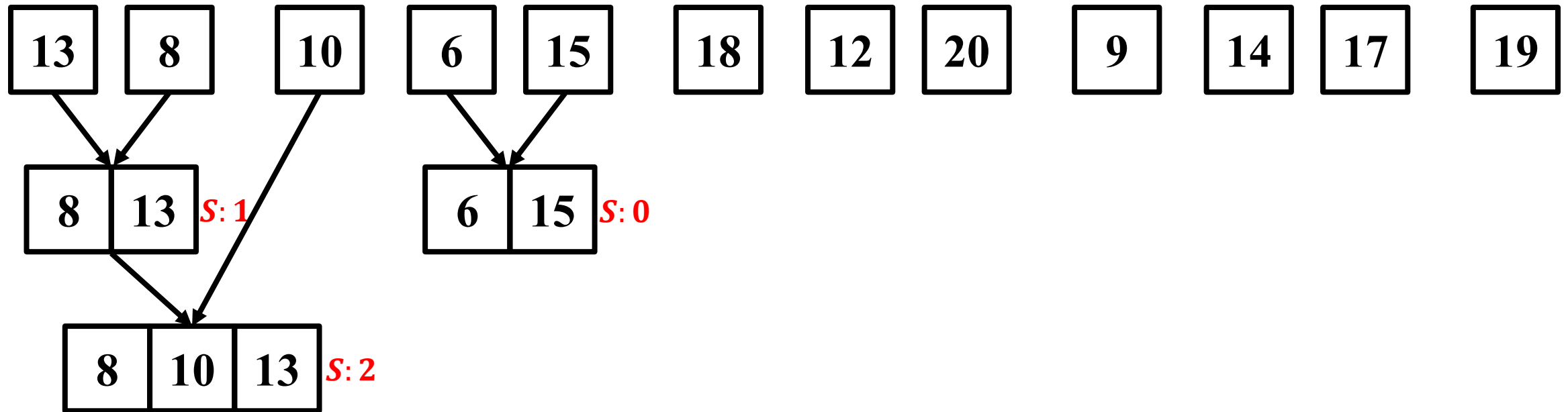


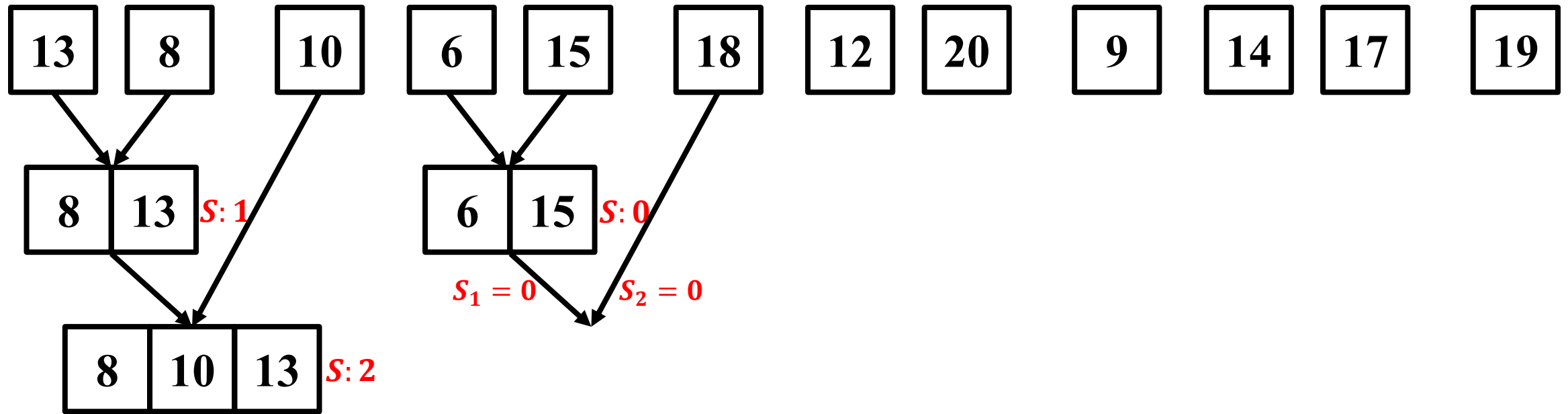


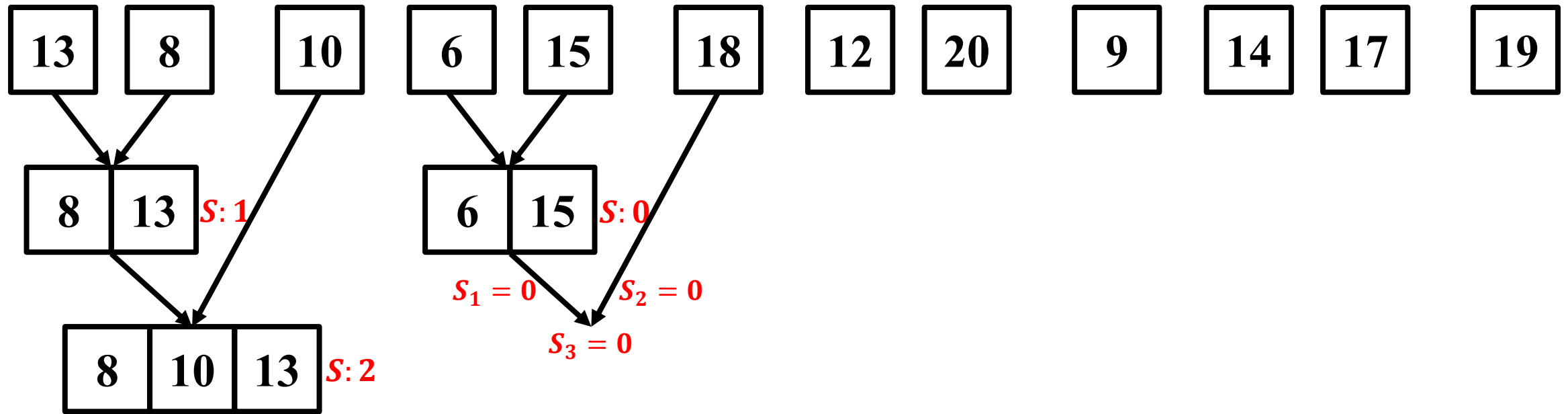




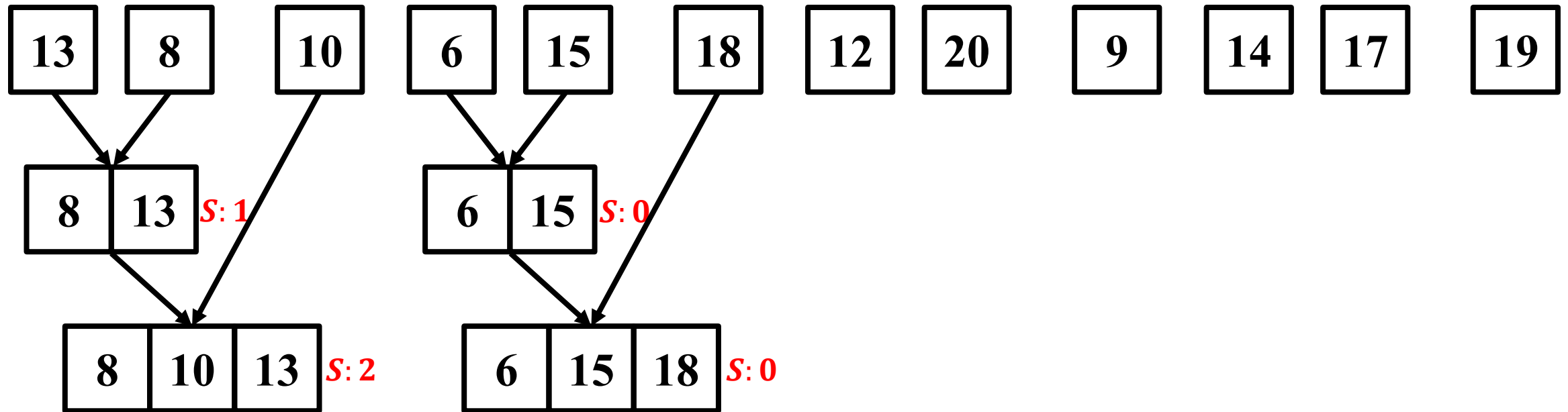


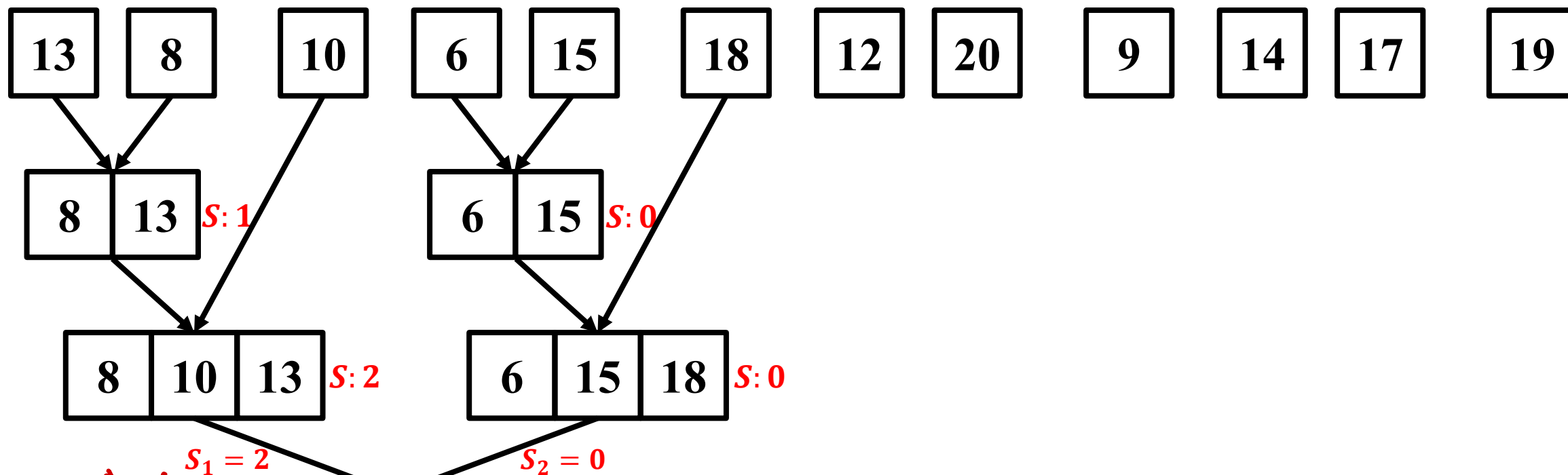




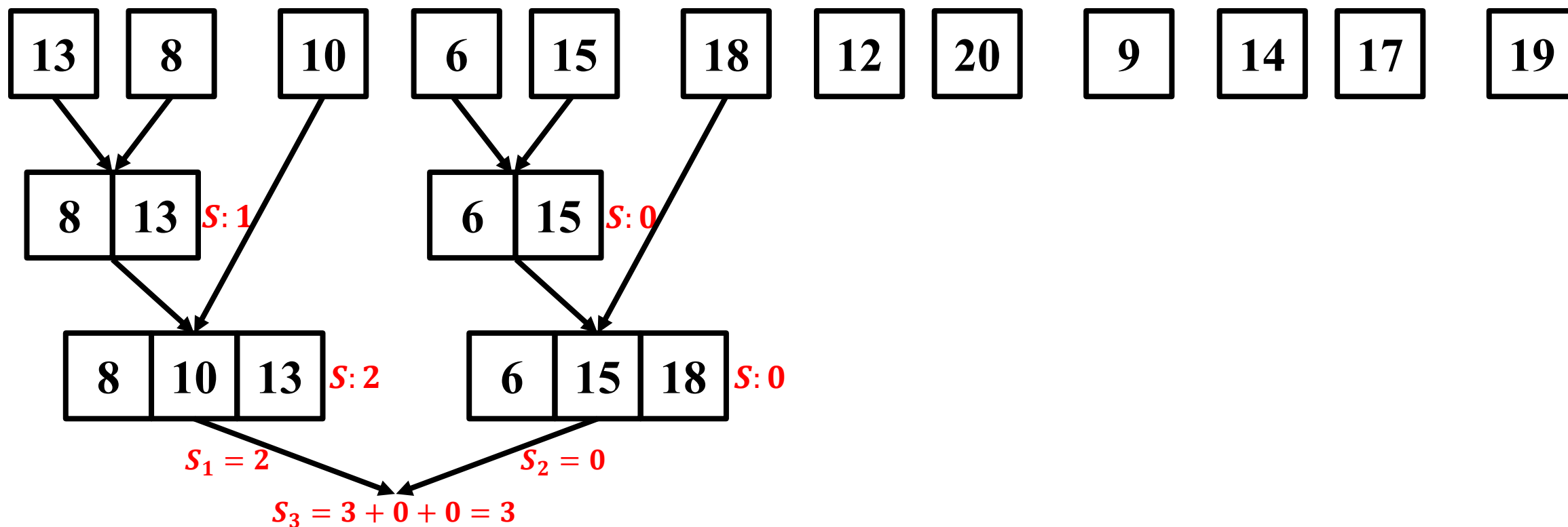


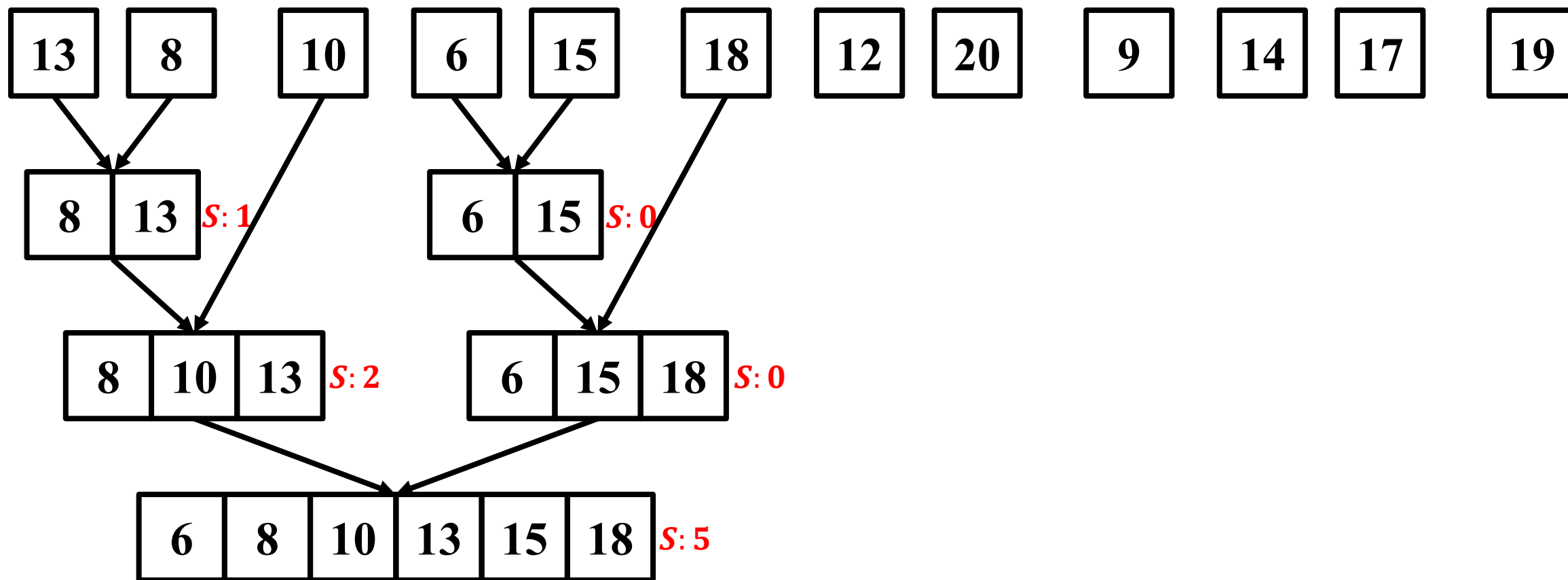
算法实例



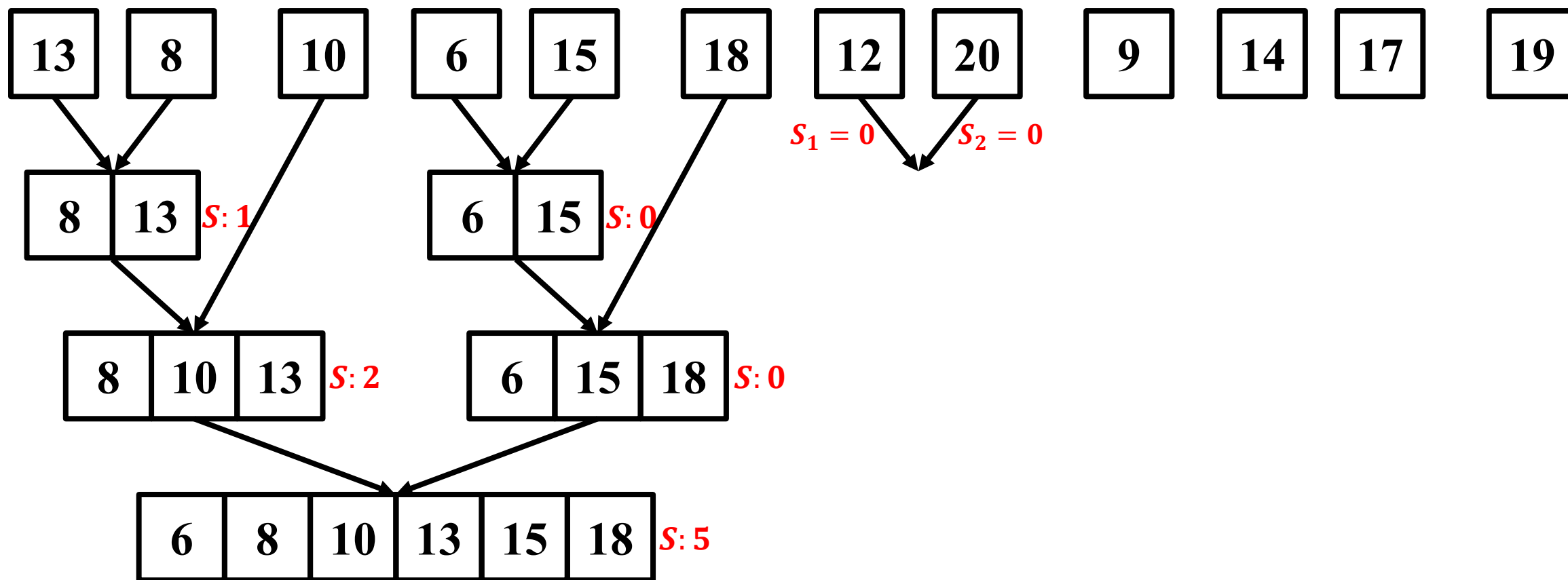


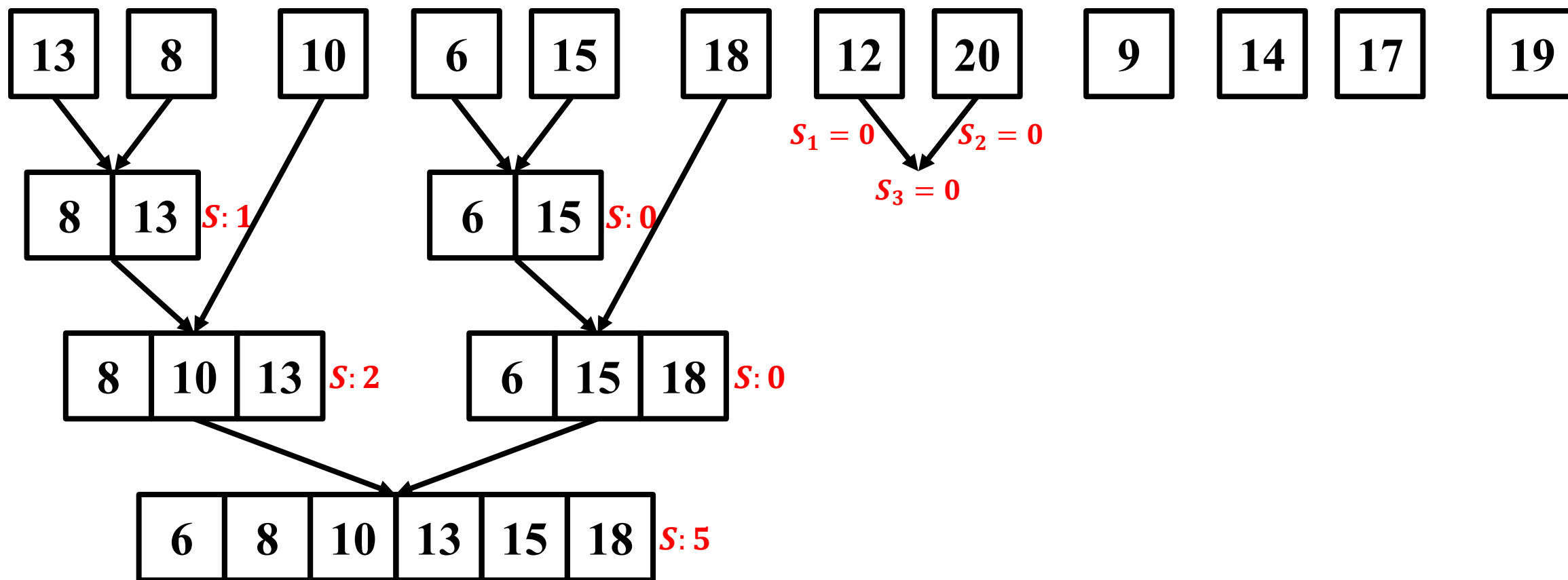
上一步求解的 $S=2$
作为这一步的输入 (直接作 S_1 的结果)
子串重排也不影响 S_1 的计算。



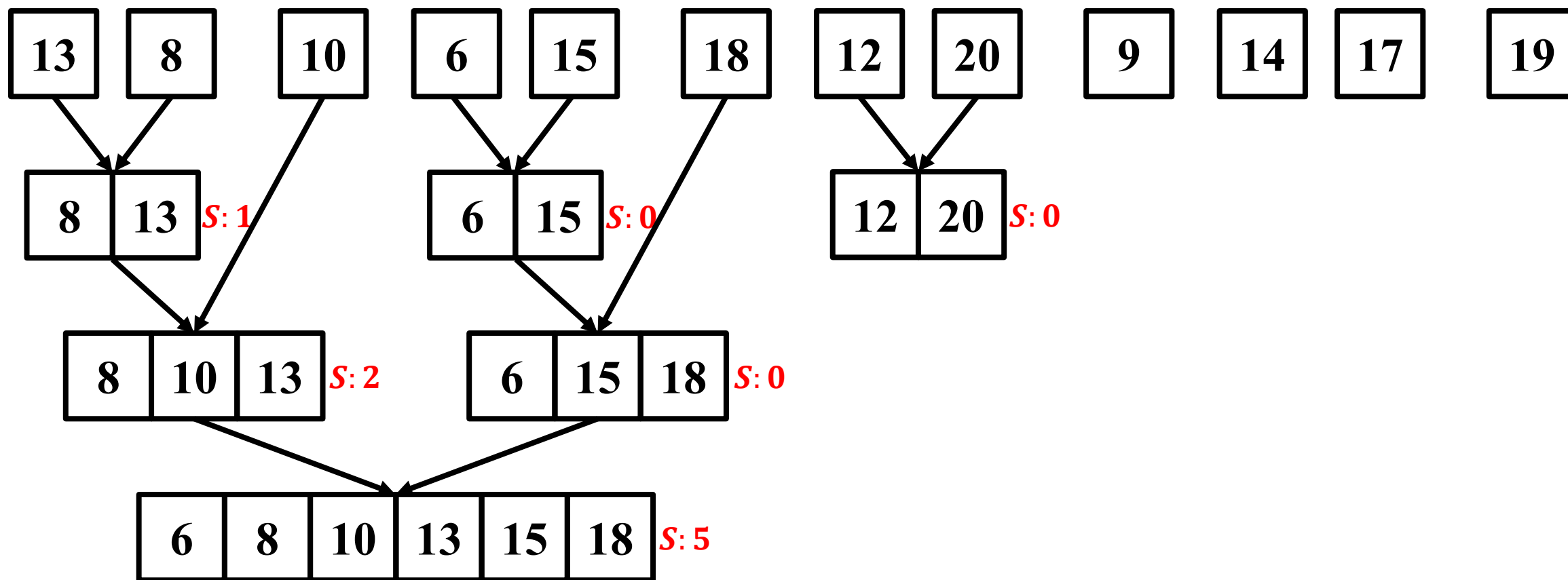


算法实例

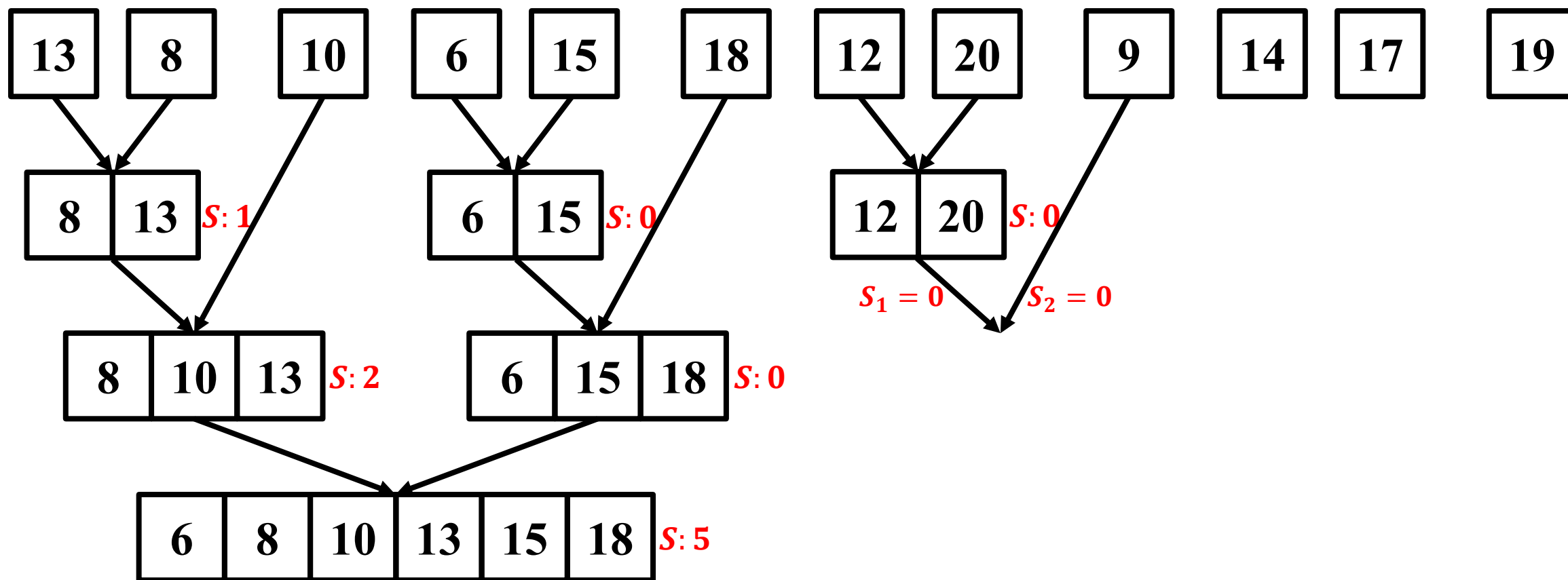


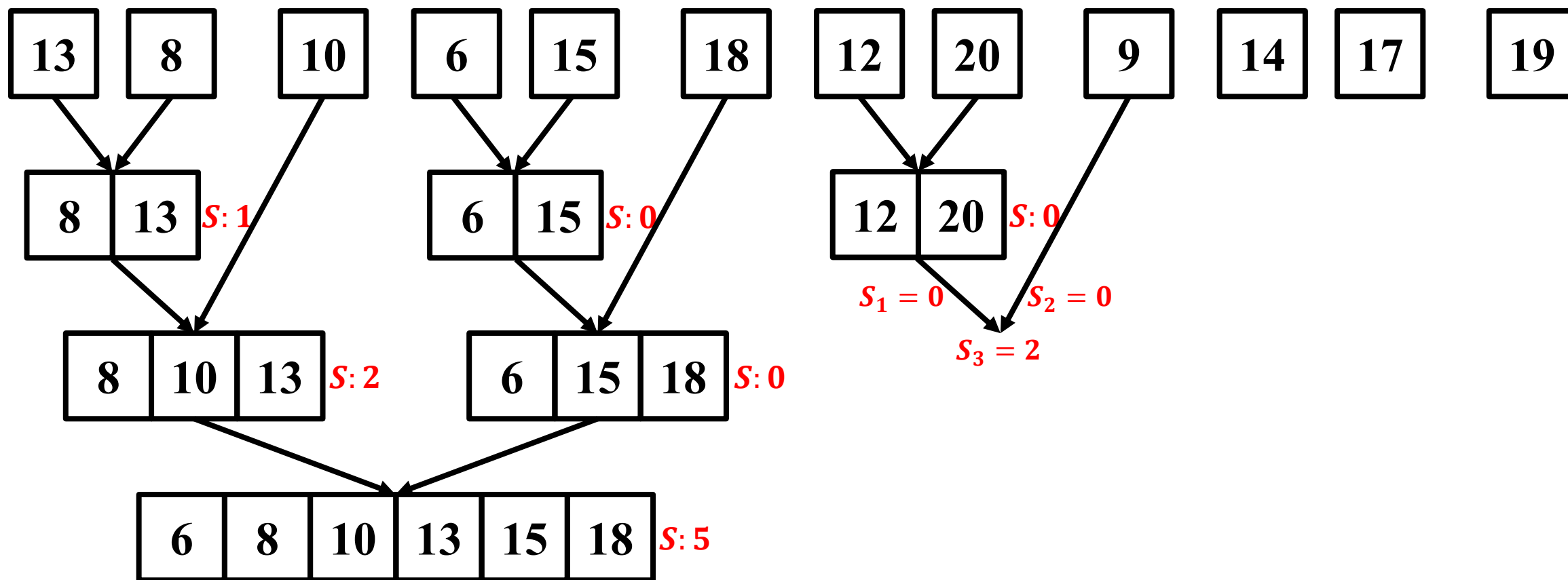


算法实例

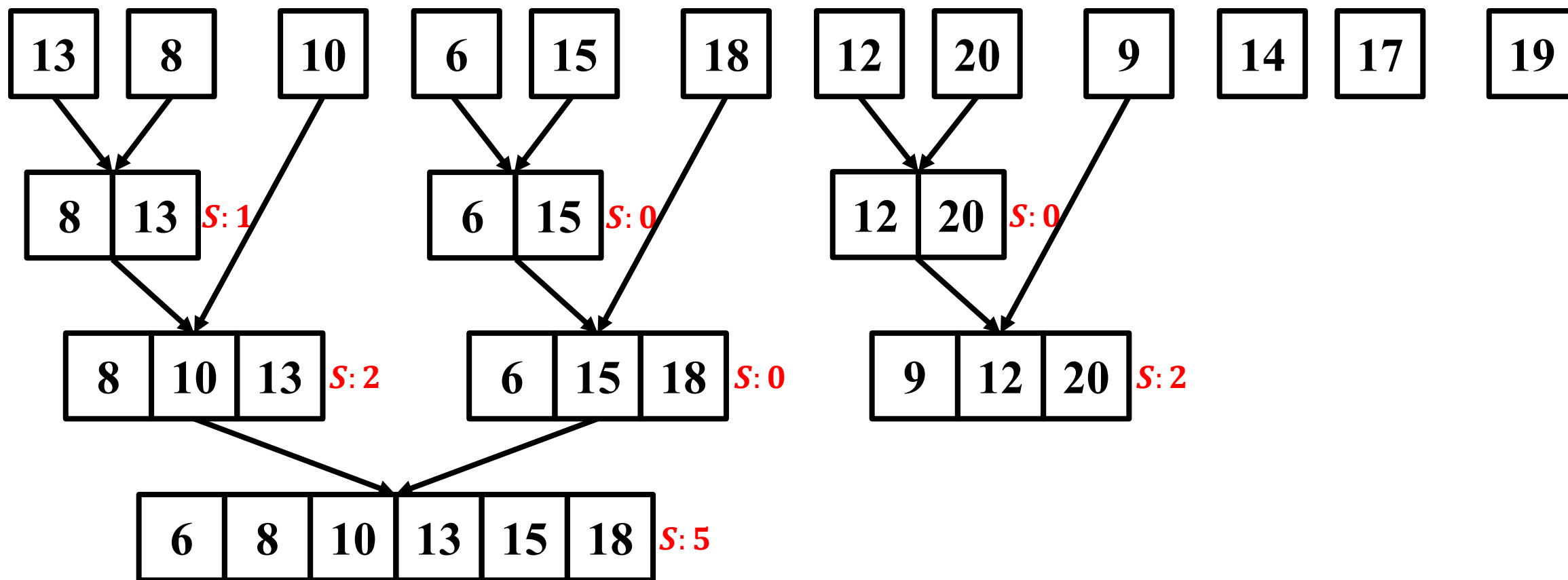


算法实例

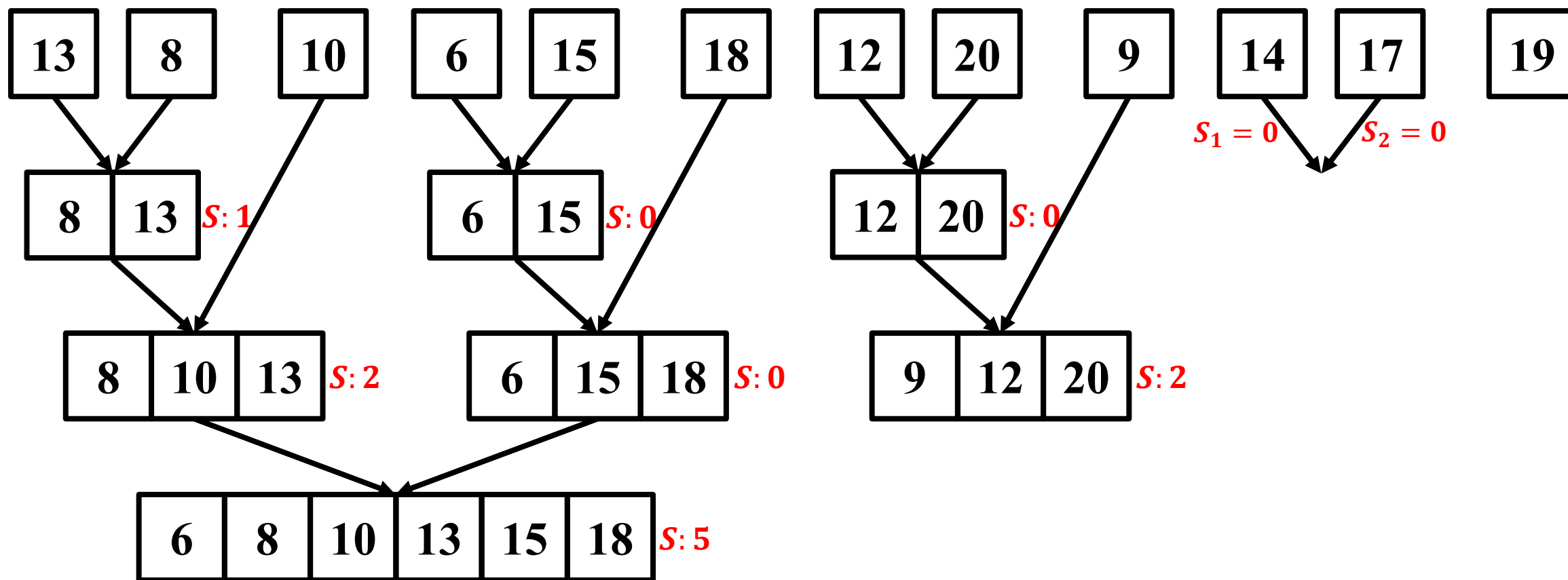




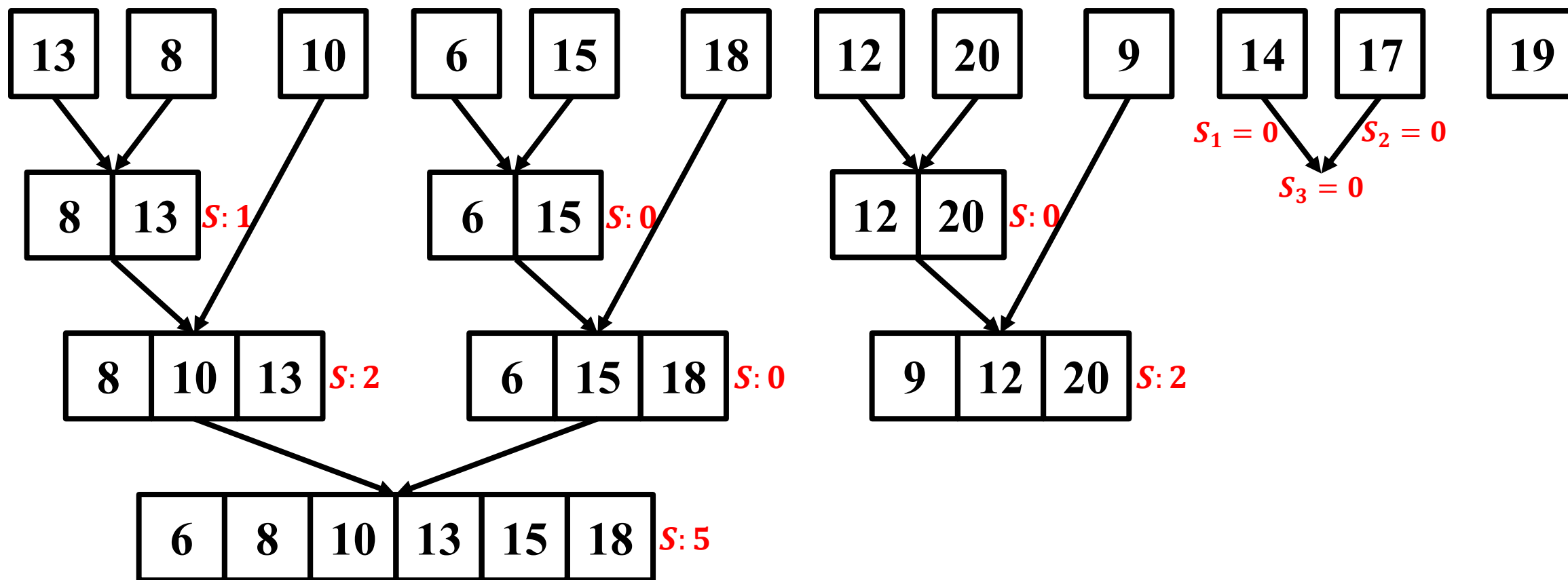
算法实例



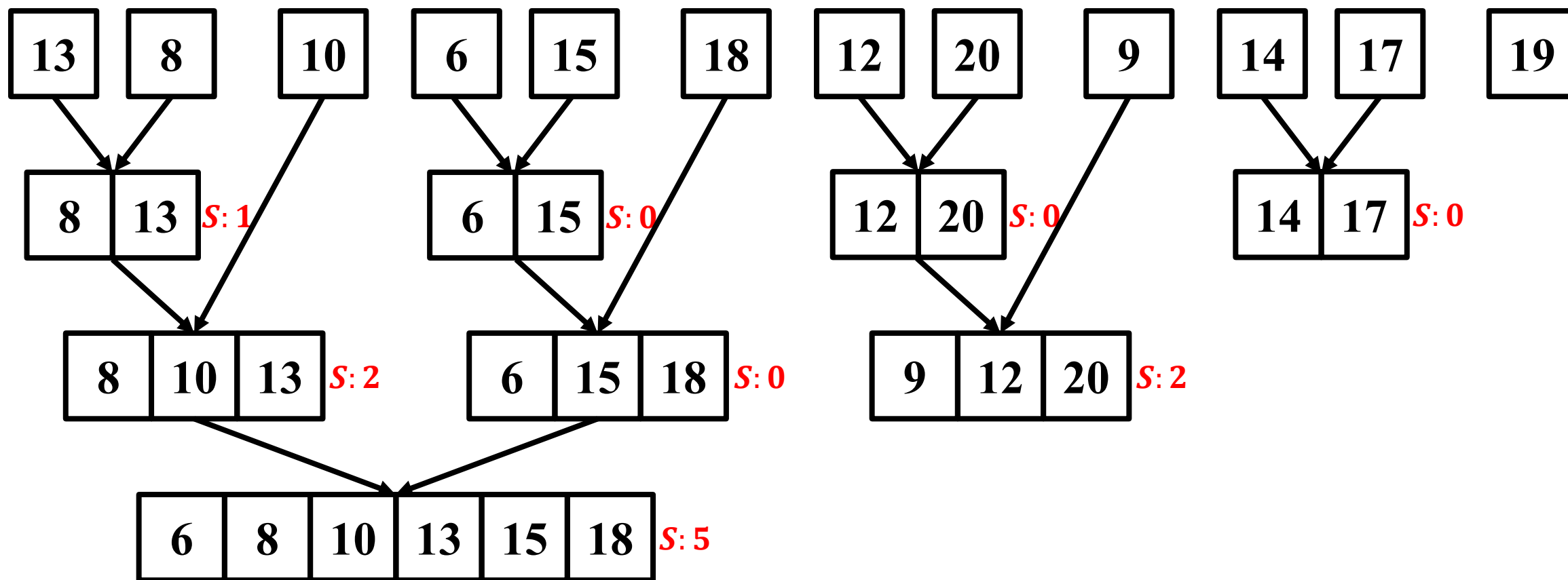
算法实例

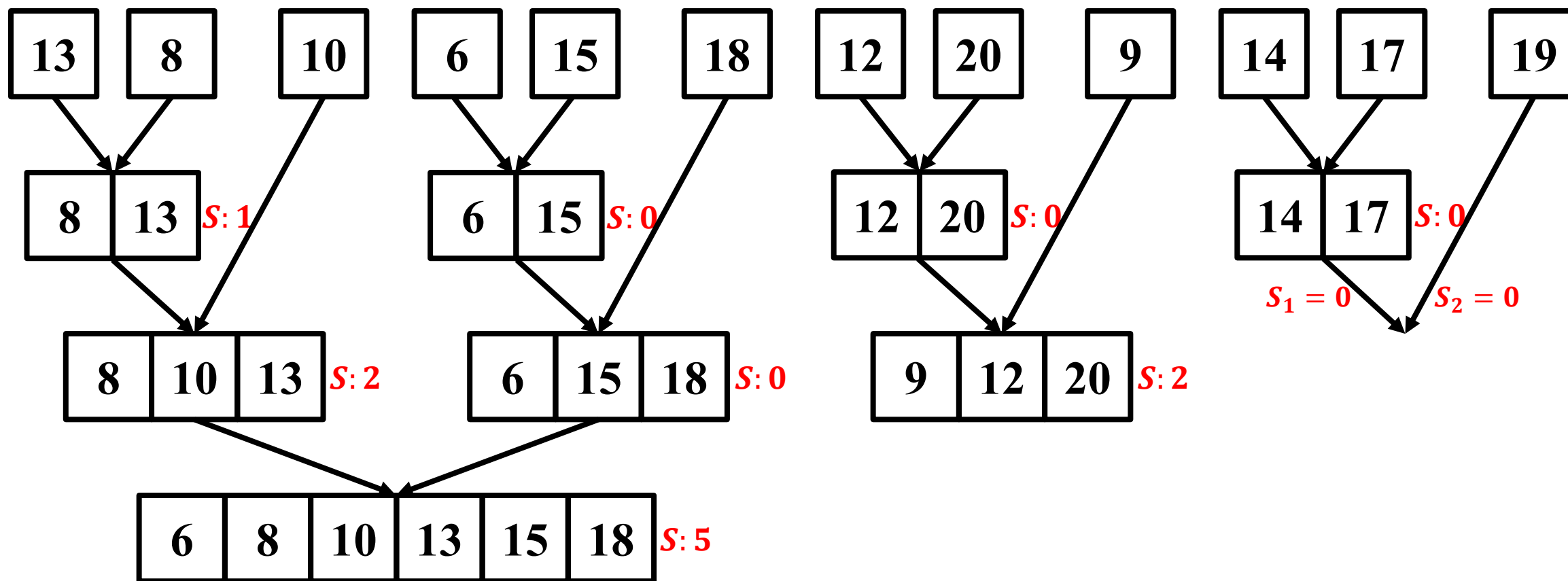


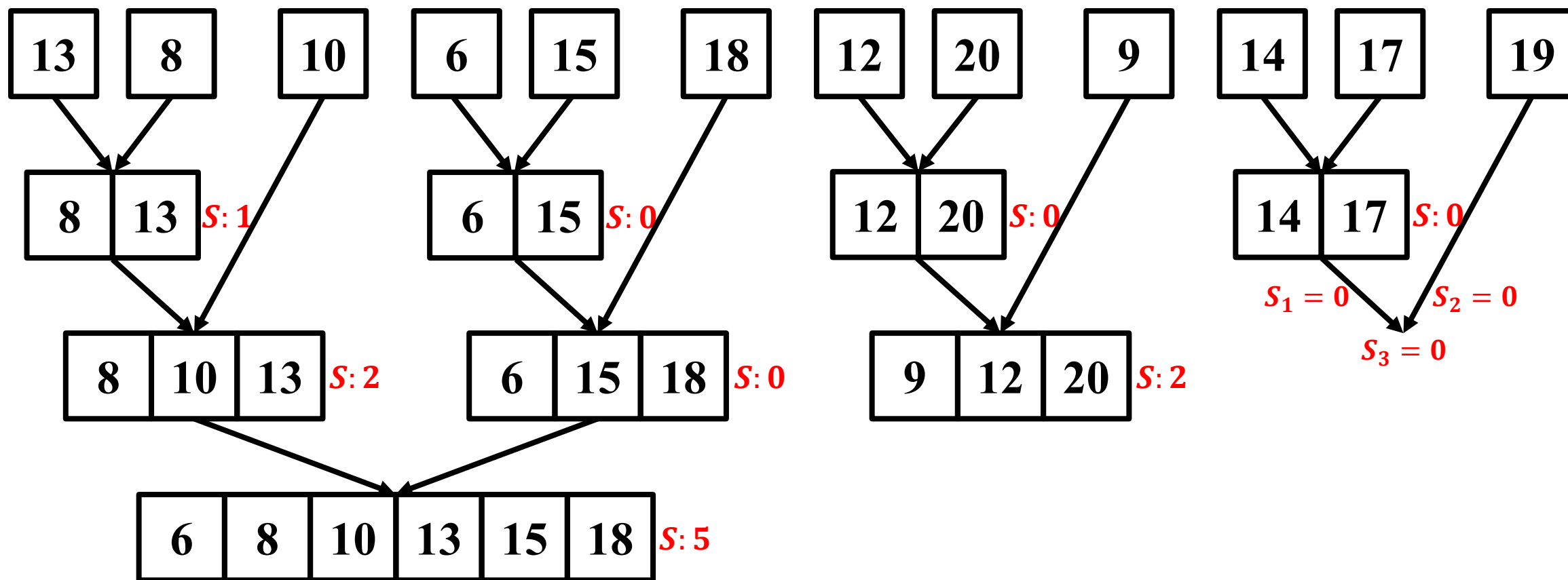
算法实例

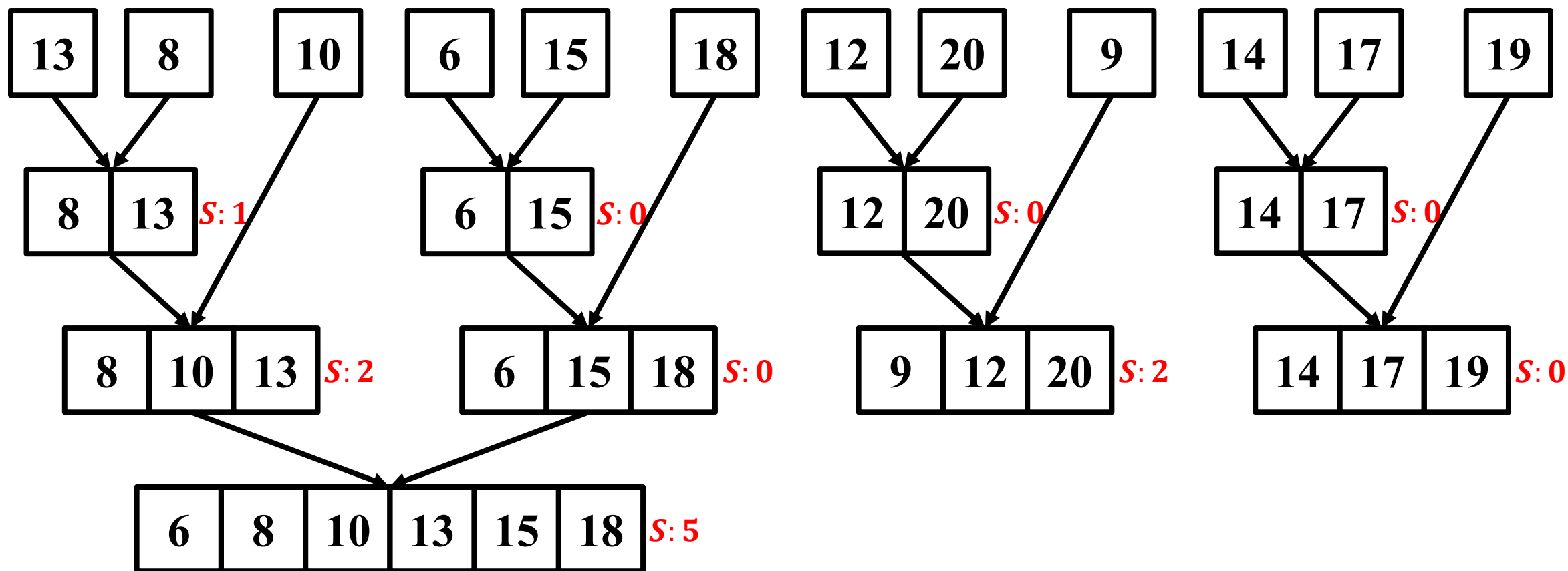


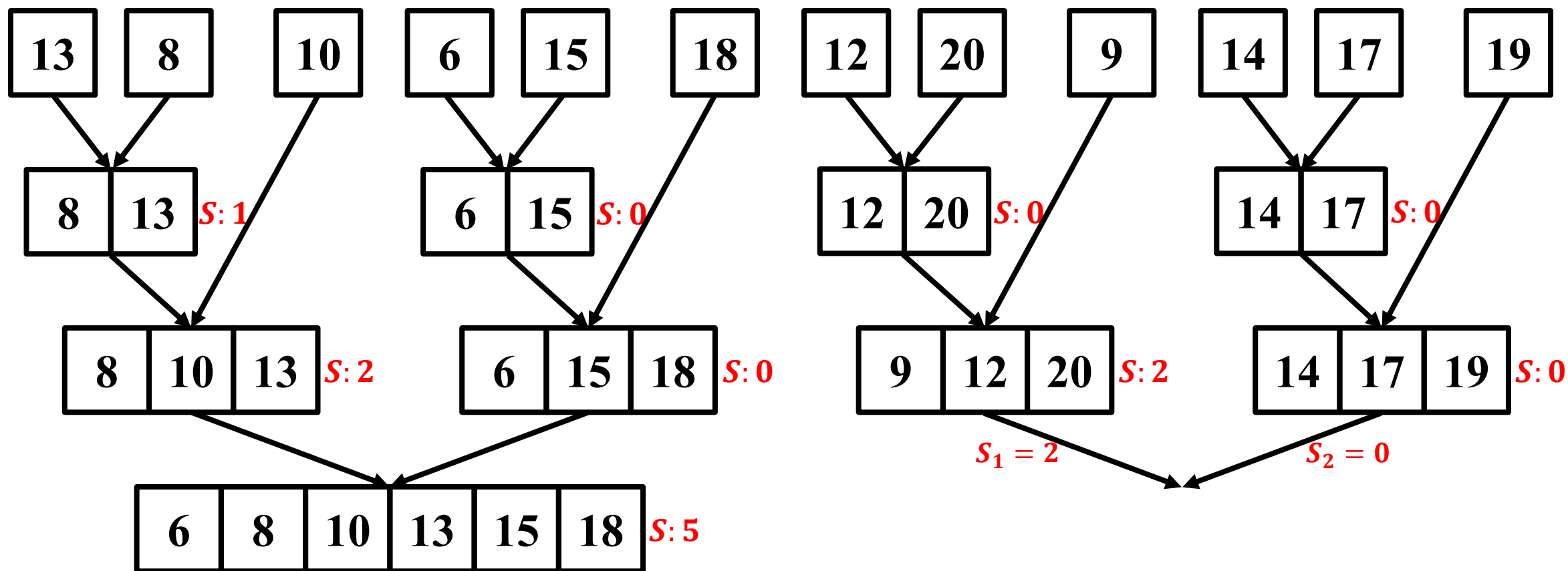
算法实例

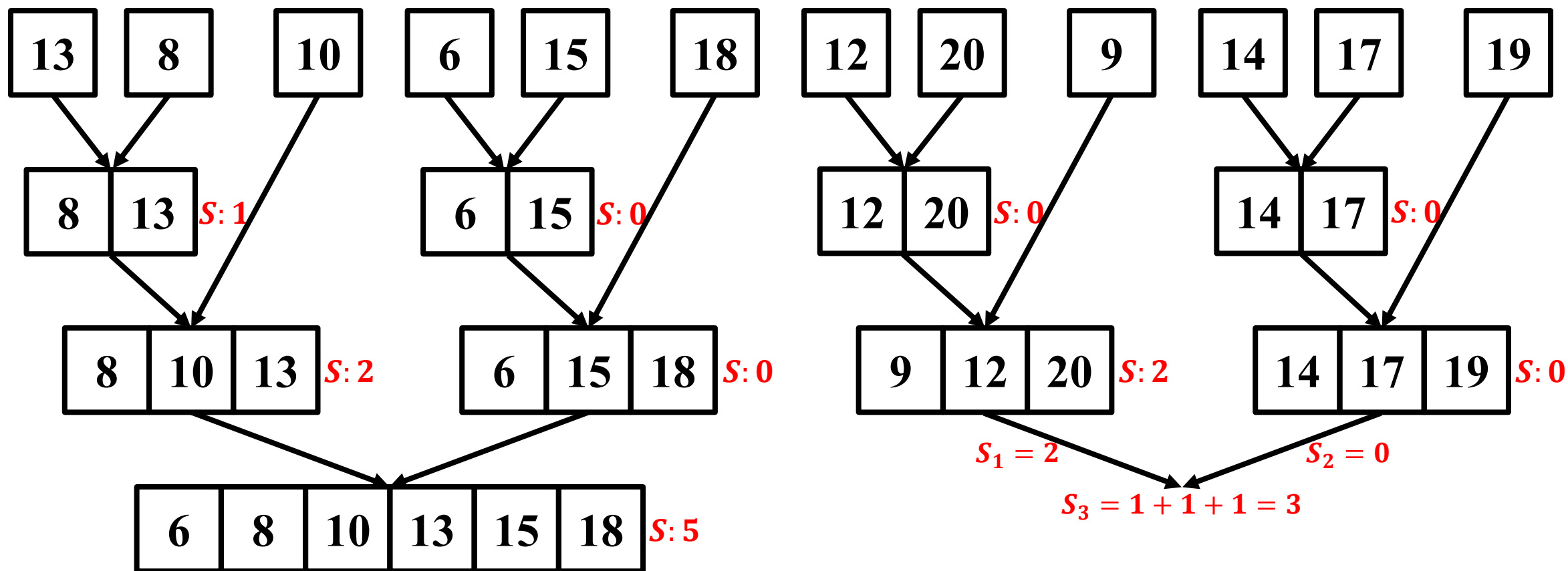


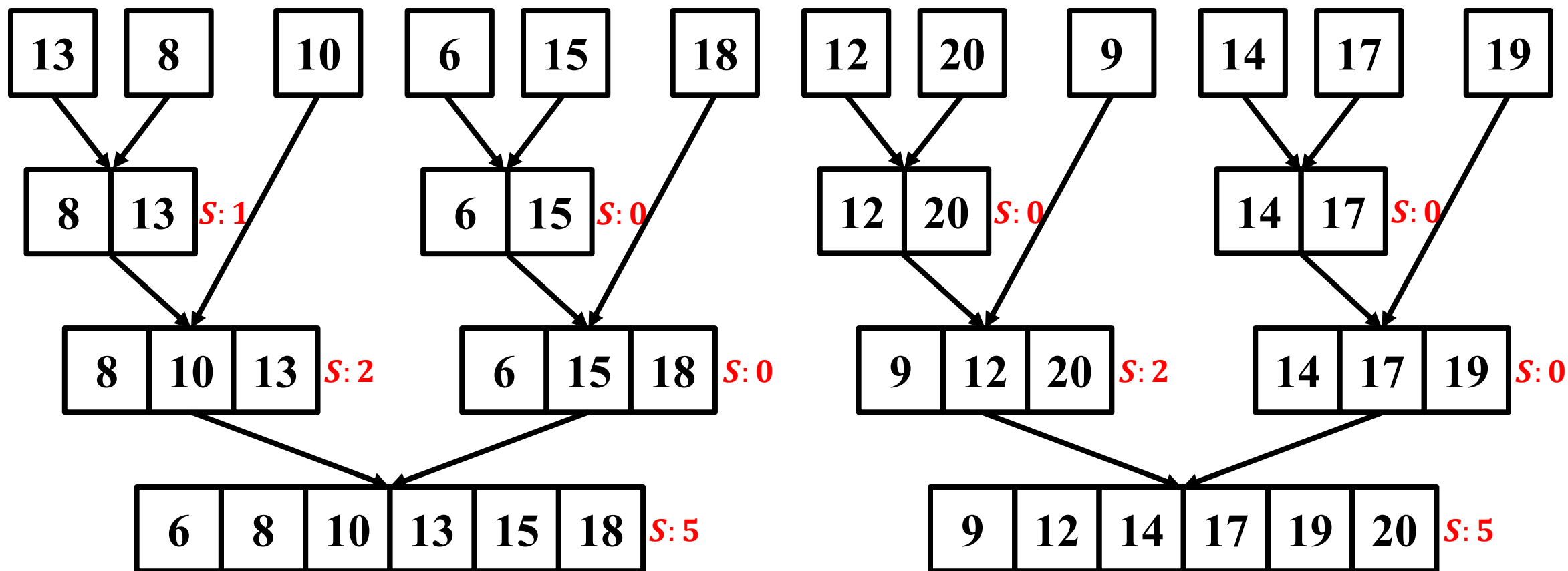


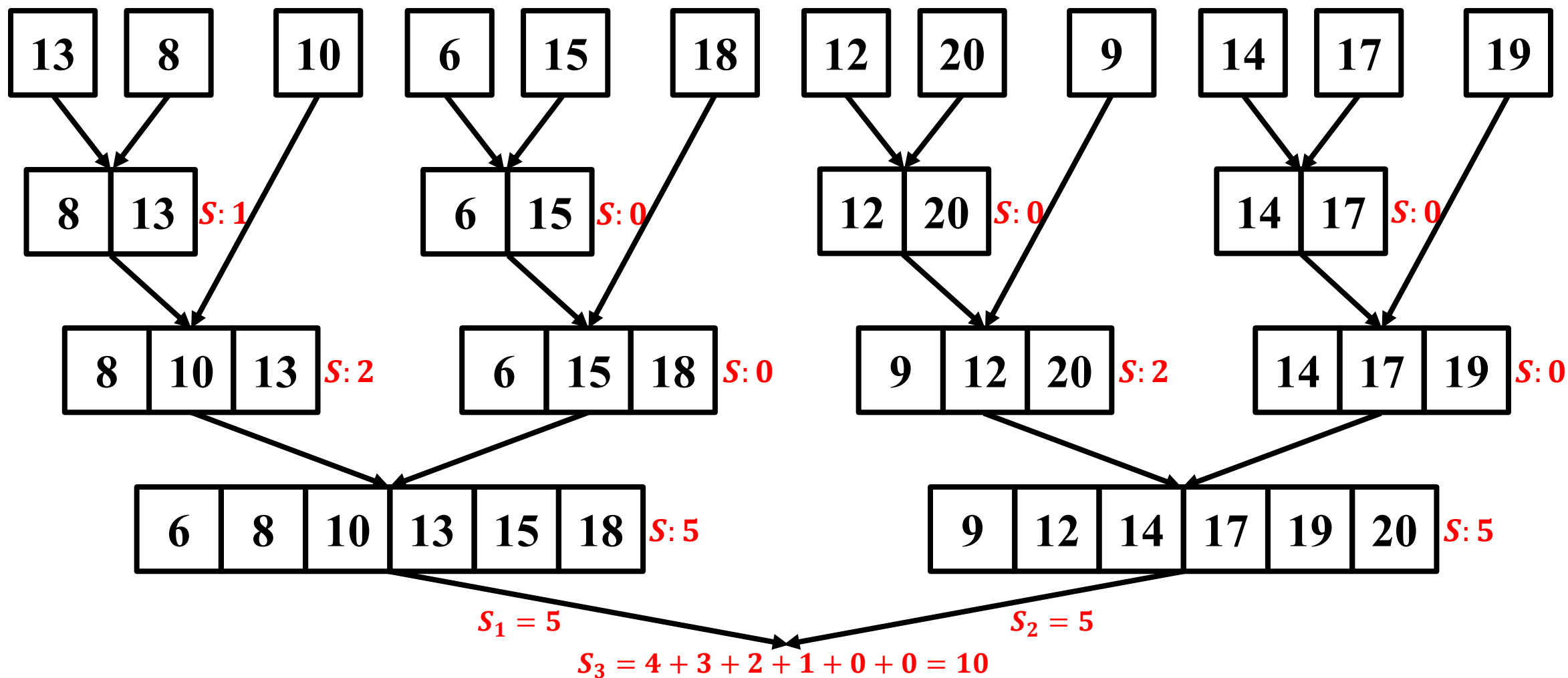




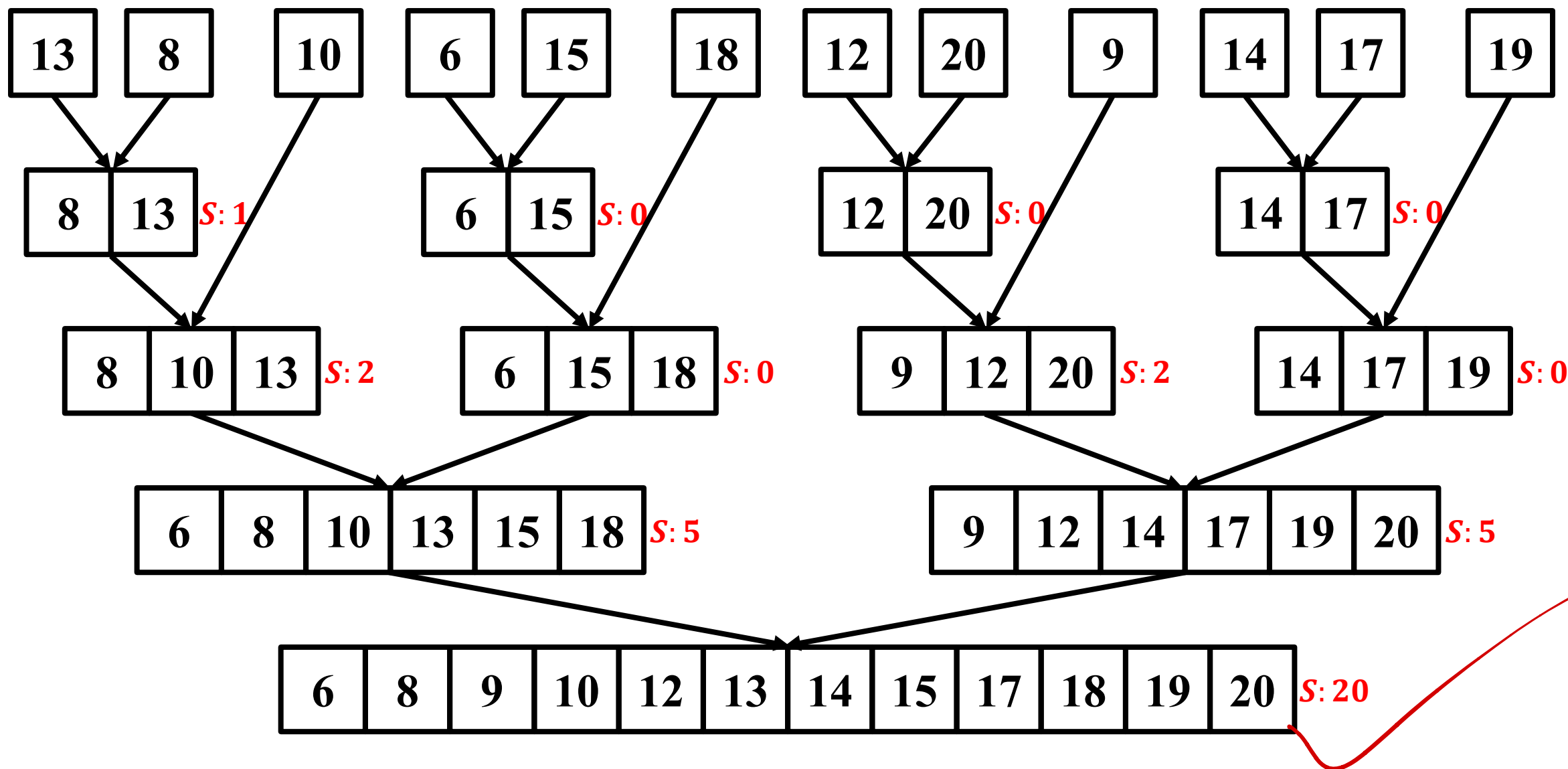








算法实例



- 归并求解: $\text{CountInver}(A, left, right)$

输入: 数组 $A[1..n]$, 数组下标 $left, right$

输出: 数组 $A[left..right]$ 的逆序对数, 递增数组 $A[left..right]$

```
if  $left \geq right$  then  
| return  $A[left..right]$   
end
```

```
 $mid \leftarrow \lfloor \frac{left+right}{2} \rfloor$ 
```

```
 $S_1 \leftarrow \text{CountInver}(A, left, mid)$ 
```

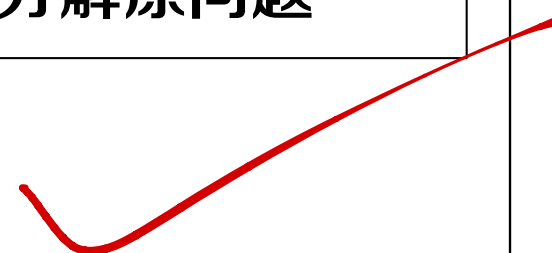
```
 $S_2 \leftarrow \text{CountInver}(A, mid + 1, right)$ 
```

```
 $S_3 \leftarrow \text{MergeCount}(A, left, mid, right)$ 
```

```
 $S \leftarrow S_1 + S_2 + S_3$ 
```

```
return  $S, A[left..right]$ 
```

分解原问题



- 归并求解: $\text{CountInver}(A, left, right)$

输入: 数组 $A[1..n]$, 数组下标 $left, right$

输出: 数组 $A[left..right]$ 的逆序对数, 递增数组 $A[left..right]$

```
if  $left \geq right$  then  
  | return  $A[left..right]$   
end
```

$mid \leftarrow \lfloor \frac{left+right}{2} \rfloor$

$S_1 \leftarrow \text{CountInver}(A, left, mid)$

$S_2 \leftarrow \text{CountInver}(A, mid + 1, right)$

$S_3 \leftarrow \text{MergeCount}(A, left, mid, right)$

$S \leftarrow S_1 + S_2 + S_3$

return $S, A[left..right]$

解决子问题

- 归并求解: $\text{CountInver}(A, left, right)$

输入: 数组 $A[1..n]$, 数组下标 $left, right$

输出: 数组 $A[left..right]$ 的逆序对数, 递增数组 $A[left..right]$

if $left \geq right$ then
| return $A[left..right]$

end

$mid \leftarrow \lfloor \frac{left+right}{2} \rfloor$

$S_1 \leftarrow \text{CountInver}(A, left, mid)$

$S_2 \leftarrow \text{CountInver}(A, mid + 1, right)$

$S_3 \leftarrow \text{MergeCount}(A, left, mid, right)$

$S \leftarrow S_1 + S_2 + S_3$

return $S, A[left..right]$

合并问题解

- 归并求解: $\text{CountInver}(A, left, right)$

初始调用: $\text{CountInver}(A, 1, n)$

```
输入: 数组  $A[1..n]$ , 数组下标  $left, right$   
输出: 数组  $A[left..right]$  的逆序对数, 递增数组  $A[left..right]$   
if  $left \geq right$  then  
    | return  $A[left..right]$   
end  
 $mid \leftarrow \lfloor \frac{left+right}{2} \rfloor$   
 $S_1 \leftarrow \text{CountInver}(A, left, mid)$   
 $S_2 \leftarrow \text{CountInver}(A, mid + 1, right)$   
 $S_3 \leftarrow \text{MergeCount}(A, left, mid, right)$   
 $S \leftarrow S_1 + S_2 + S_3$   
return  $S, A[left..right]$ 
```


时间复杂度分析



- 输入规模为: n

- $$T(n) = \begin{cases} 1, & n = 1 \\ 2 \cdot T(n/2) + O(n), & n > 1 \end{cases}$$

输入: 数组 $A[1..n]$, 数组下标 $left, right$

输出: 数组 $A[left..right]$ 的逆序对数, 递增数组 $A[left..right]$

```
if  $left \geq right$  then  
| return  $A[left..right]$   
end
```

```
 $mid \leftarrow \lfloor \frac{left+right}{2} \rfloor$ 
```

```
 $S_1 \leftarrow \text{CountInver}(A, left, mid)$ 
```

```
 $S_2 \leftarrow \text{CountInver}(A, mid + 1, right)$ 
```

```
 $S_3 \leftarrow \text{MergeCount}(A, left, mid, right)$ 
```

```
 $S \leftarrow S_1 + S_2 + S_3$ 
```

```
return  $S, A[left..right]$ 
```

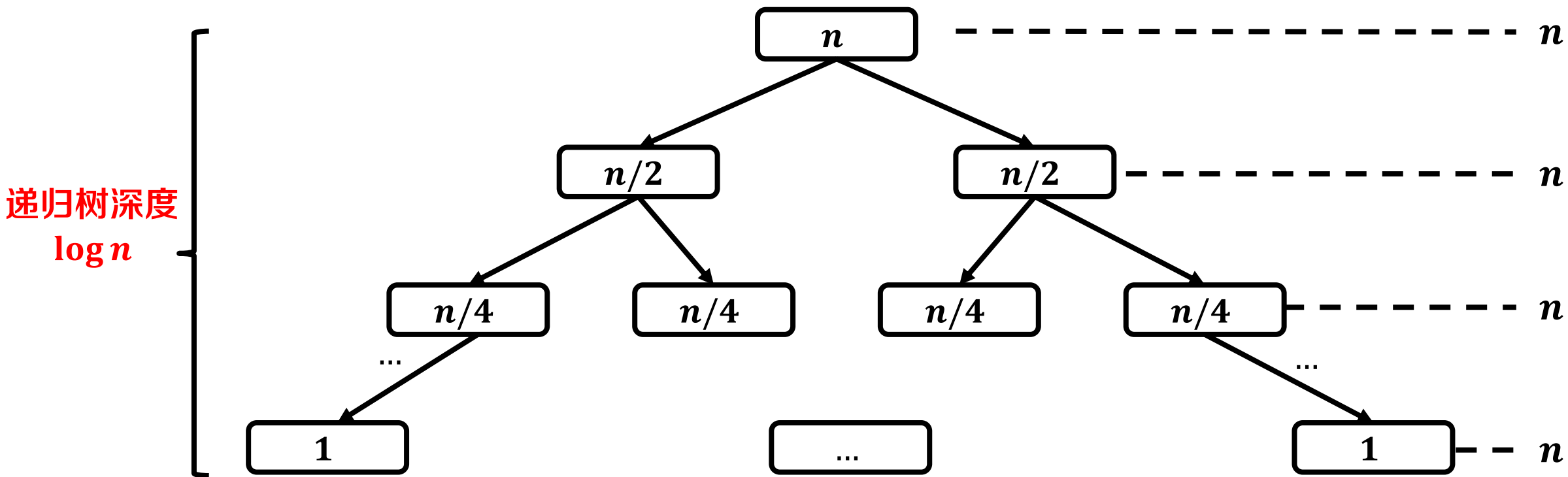
} $2 \cdot T(n/2)$
 $O(n)$

时间复杂度分析



- 输入规模为: n

- $$T(n) = \begin{cases} 1, & n = 1 \\ 2 \cdot T(n/2) + O(n), & n > 1 \end{cases}$$



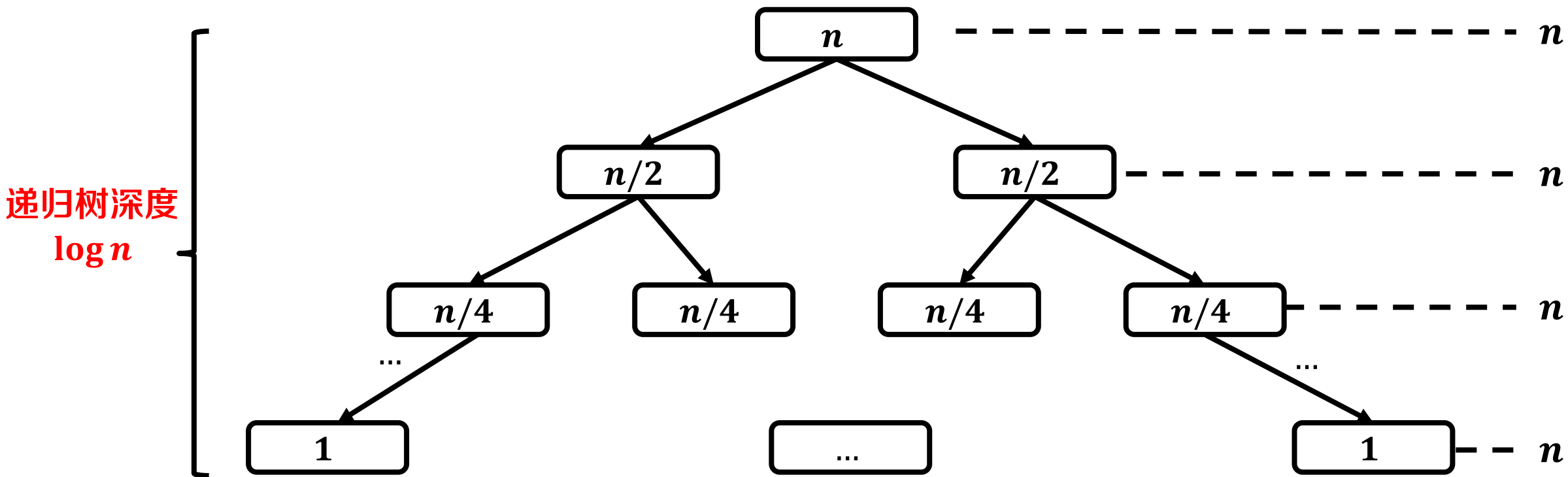
时间复杂度分析



- 输入规模为: n

- $$T(n) = \begin{cases} 1, & n = 1 \\ 2 \cdot T(n/2) + O(n), & n > 1 \end{cases}$$

$$T(n) = O(n \log n)$$



- 在本问题中，我们设计了四个算法：

算法名称	合并求解复杂度	时间复杂度
蛮力枚举	—	$O(n^2)$
分而治之+直接计算	$O(n^2)$	$O(n^2)$
分而治之+排序求解	$O(n \log n)$	$O(n \log^2 n)$
分而治之+归并求解	$O(n)$	$O(n \log n)$

小结



- 在本问题中，我们设计了四个算法：

算法名称	合并求解复杂度	时间复杂度
蛮力枚举	—	$O(n^2)$
分而治之+直接计算	$O(n^2)$	$O(n^2)$
分而治之+排序求解	$O(n \log n)$	$O(n \log^2 n)$
分而治之+归并求解	$O(n)$	$O(n \log n)$

问题：分治策略关键是什么？

答案：合理设计合并求解算法