算法分析与设计 第二讲

排序算法及分析

本次课主要内容

- 排序问题
- 插入排序
- 合并排序
- 递归式
- 算法分析

排序问题

Input: sequence $\langle a_1, a_2, \ldots, a_n \rangle$ of numbers.

Output: permutation $\langle a'_1, a'_2, \ldots, a'_n \rangle$ Such that $a'_1 \leq a'_2 \leq \ldots \leq a'_n$.

Example:

Input: 8 2 4 9 3 6

Output: 2 3 4 6 8 9

排序算法

- 插入排序
- 合并排序
- 冒泡排序
- 堆排序
- 快速排序
- 选择排序
- ……

插入排序（INSERTION-SORT）

INSERTION-SORT (A, n) ⊳ $A[1 \,..\, n]$

for $j \leftarrow 2$ to n

do $key \leftarrow A[j]$

$i \leftarrow j - 1$

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

do $A[i+1] \leftarrow A[i]$

$i \leftarrow i - 1$

$A[i+1] = key$

"pseudocode"

插入排序（INSERTION-SORT）

INSERTION-SORT (A, n) ⊳ $A[1\,.\,.\,n]$

for $j \leftarrow 2$ to n

do $key \leftarrow A[j]$

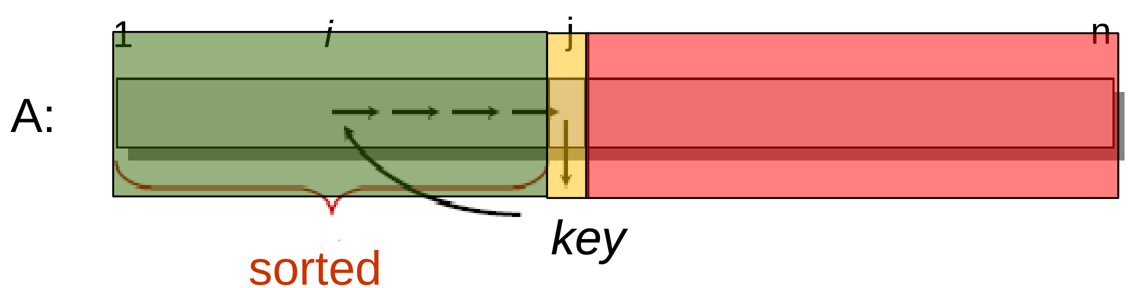
$i \leftarrow j-1$

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

do $A[i+1] \leftarrow A[i]$

$i \leftarrow i-1$

$A[i+1] = key$



插入排序举例

8　　2　　4　　9　　3　　6

插入排序举例

8　　2　　4　　9　　3　　6

插入排序举例

8	2	4	9	3	6
2	8	4	9	3	6

插入排序举例

8　2　4　9　3　6

2　8　4　9　3　6

插入排序举例

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

插入排序举例

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

插入排序举例

8	2	4	9	3	6
2	8	4	9	3	6
2	4	8	9	3	6
2	4	8	9	3	6

插入排序举例

8　2　4　9　3　6

2　8　4　9　3　6

2　4　8　9　3　6

2　4　8　9　3　6

插入排序举例

8	2	4	9	3	6
2	8	4	9	3	6
2	4	8	9	3	6
2	4	8	9	3	6
2	3	4	8	9	6

插入排序举例

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

2 4 8 9 3 6

2 3 4 8 9 6

插入排序举例

8 2 4 9 3 6

2 8 4 9 3 6

2 4 8 9 3 6

2 4 8 9 3 6

2 3 4 8 9 6

2 3 4 6 8 9 *done*

插入排序的分析

- 插入排序的时间开销
  - 与输入规模有关
  - 与输入序列的特性有关

插入排序的分析

- 最佳情况运行时间
  - 输入数组已经排好序
- 最坏情况运行时间
  - 问题要求最终按递增的顺序排列
  - 但输入数组按递减顺序排列

插入排序的分析

```
INSERTION-SORT (A, n)
    for j ←2 to n
      do key ← A[ j]
         ⊳ Insert A[ j]
         i ← j –1
         while i > 0 and A[i] > key
                  do A[i+1] ← A[i]
                     i ← i –1
         A[i+1] = key
```

插入排序的分析

- 为分析做的简化
  - 忽略每条语句的真实代价
  - 只考虑最高次项
  - 忽略最高次项的系数
- 插入排序的最坏情况时间复杂度
  - $\Theta(n^2)$

插入排序的C++示例代码

```
void InsertSort(int * Array,int Size)
{
        int j,t;
        for(int i = 1;i < Size;i++)
        {
                for( j = 0;j < i;j++)
                {
                        if(Array[j] > Array[i])
                                break;
                }
                t = Array[i];
                for(int k = i-1 ;k >= j;k--)
                {
                        Array[k+1] = Array[k];
                }
                Array[j] = t;
        }
}
```

插入排序示例

- 请对“3 23 25 8 1 23 16 15”应用插入排序算法进行排序，并给出排序过程

合并排序（Merge Sort）

MERGE-SORT $A[1 \;.\;.\; n]$

1. If $n = 1$, done.
2. Recursively sort $A[\;1 \;.\;.\;.\; n/2.]$ and
   $A[\;[n/2]+1 \;.\;.\; n\;]$.
3. “Merge” the 2 sorted lists.

Key subroutine: MERGE

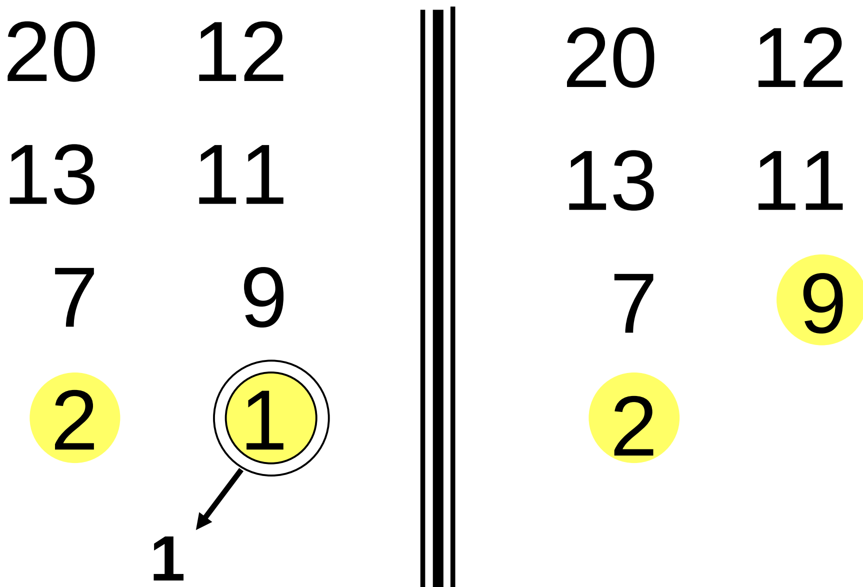
Merging two sorted arrays

20	12
13	11
7	9
2	1

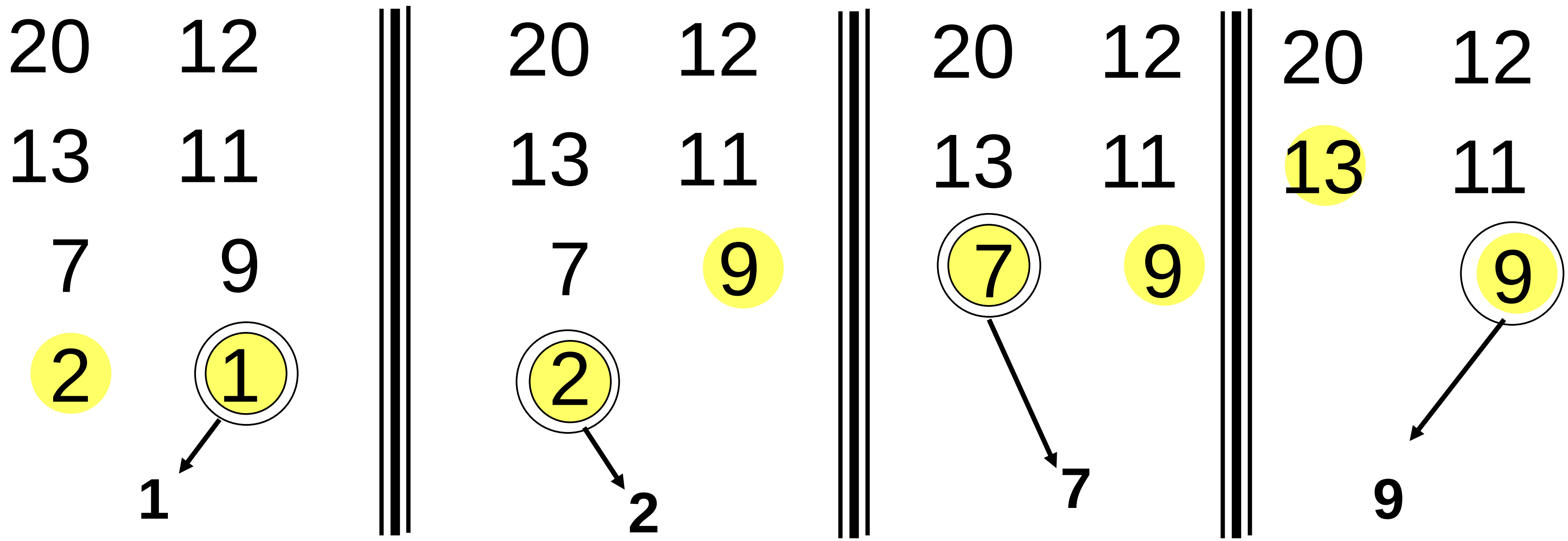
20	12
13	11
7	9
2	1

1

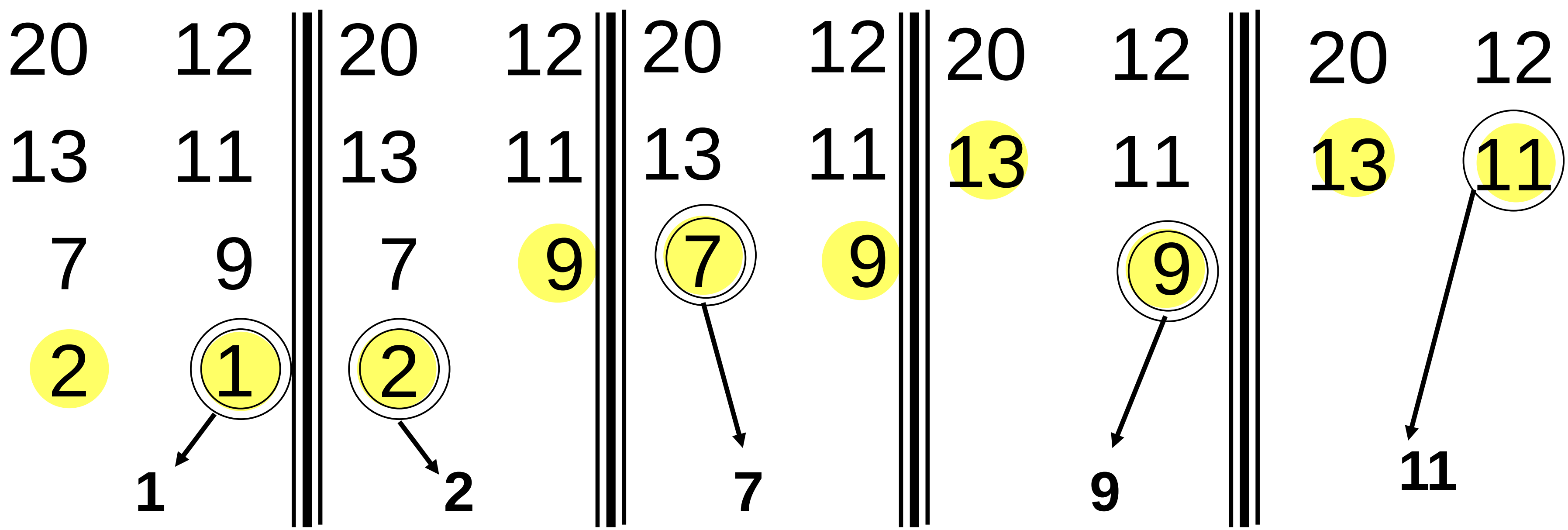
Merging two sorted arrays



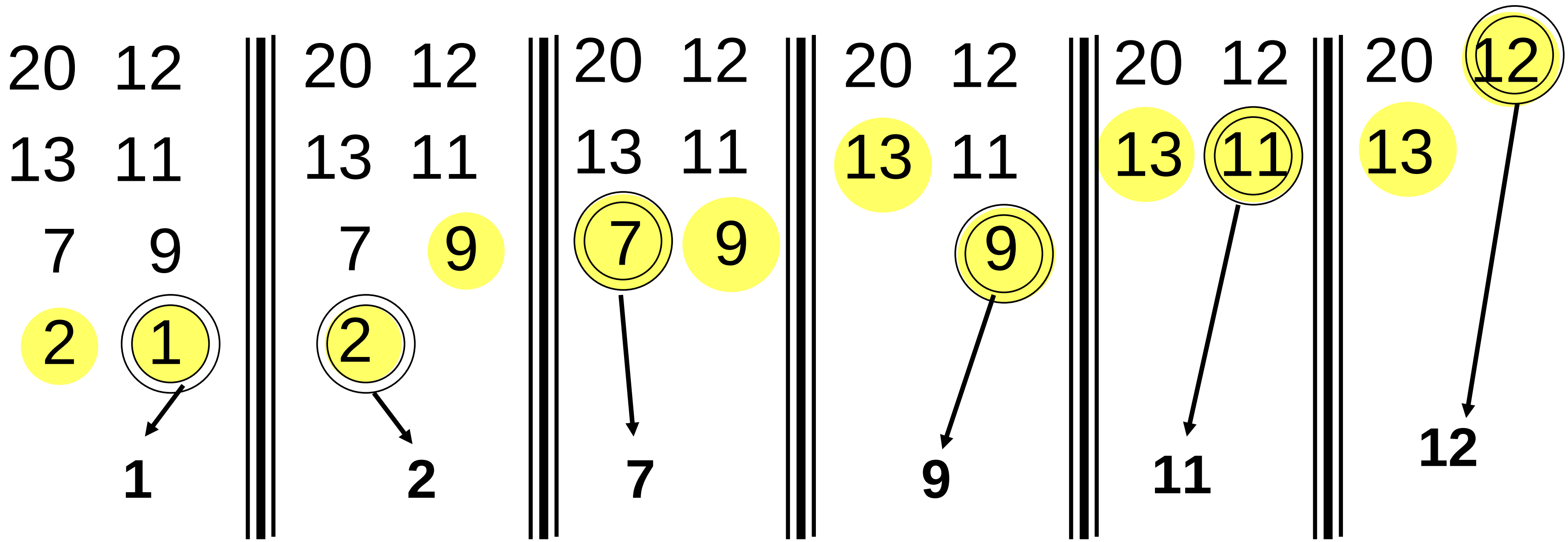
Merging two sorted arrays



Merging two sorted arrays



Merging two sorted arrays



Time = $\Theta(n)$ to merge a total of n elements (linear time).

合并排序举例

- 5，2，4，7，1，3，2，6

合并排序分析

T(n)	
	MERGE-SORTA[1 . . n]
Θ(1)	1.If n= 1, done.
2T(n/2)	2.Recursively sort A[1 . . ⌈n/2⌉] and A[⌈n/2⌉+1 . . n] .
Θ(n)	3.“Merge”the 2sorted lists

Recursion tree

- Solve $T(n) = 2T(n/2) + cn$, where $c > 0$ is constant.
- $\Theta(n \lg n)$
- $\Theta(n \lg n)$ grows more slowly than $\Theta(n^2)$
- Merge sort asymptotically beats insertion sort in the worst case
- In practice, merge sort beats insertion sort for n> 30 or so

渐近记号

- O
- Ω
- Θ

O

- 上界

$f(n) = O(g(n))$, 存在$c>0$, $n_0>0$， 使得对所有 $n \geq n_0$， 都有 $0 \leq f(n) \leq cg(n)$

- $2n^2 = O(n^3)$?
  - $c = 1$, $n_0 = 2$
- $6n^3 = O(n^2)$? ✗
- $f(n) = n^3 + O(n^2)$?

Ω

- 下界
- $n^2=\Omega(n^2)$?
- $n^2=\Omega(n)$?
- $n^2=\Omega(\lg n)$?

Θ

- 确界

- $n^2 - 2n = \Theta(n^2)$？