

向量

起泡排序



通过感觉，物体会一一呈现在我们的面前，就像在自然中一样；
通过比较，我重新安排或者调整它们的顺序。

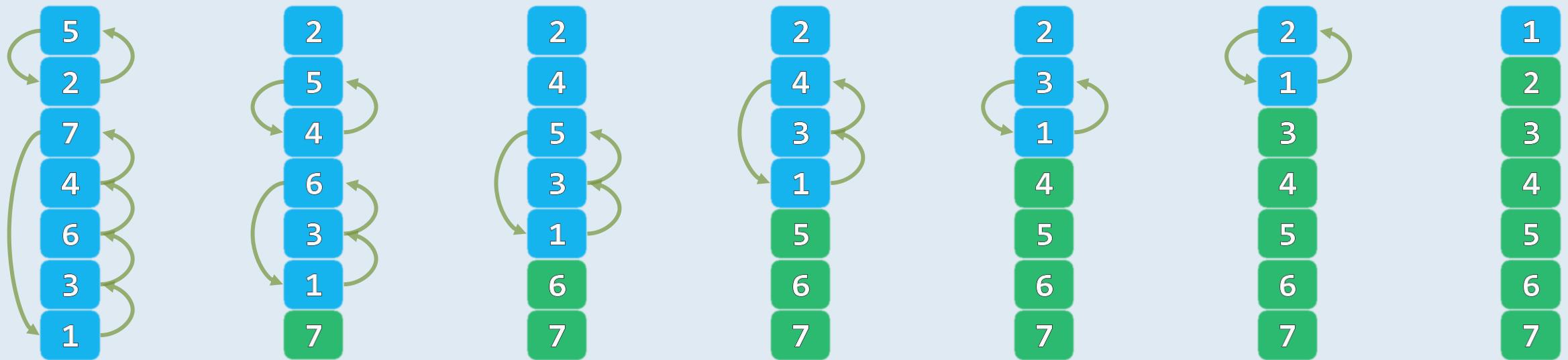
邓俊辉

deng@tsinghua.edu.cn

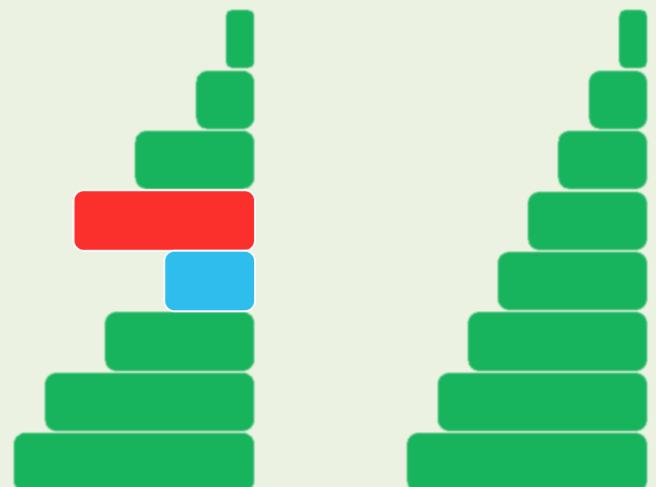
排序器：统一入口

```
❖ template <typename T> void Vector<T>::sort( Rank lo, Rank hi ) {  
    switch ( rand() % 6 ) {  
        case 1: bubbleSort( lo, hi ); break; //起泡排序  
        case 2: selectionSort( lo, hi ); break; //选择排序(习题)  
        case 3: mergeSort( lo, hi ); break; //归并排序  
        case 4: heapSort( lo, hi ); break; //堆排序(第12章)  
        case 5: quickSort( lo, hi ); break; //快速排序(第14章)  
        default: shellSort( lo, hi ); break; //希尔排序(第14章)  
    } //随机选择算法以充分测试。应用时可视具体问题的特点，灵活确定或扩充  
} //向量区间[lo, hi)排序
```

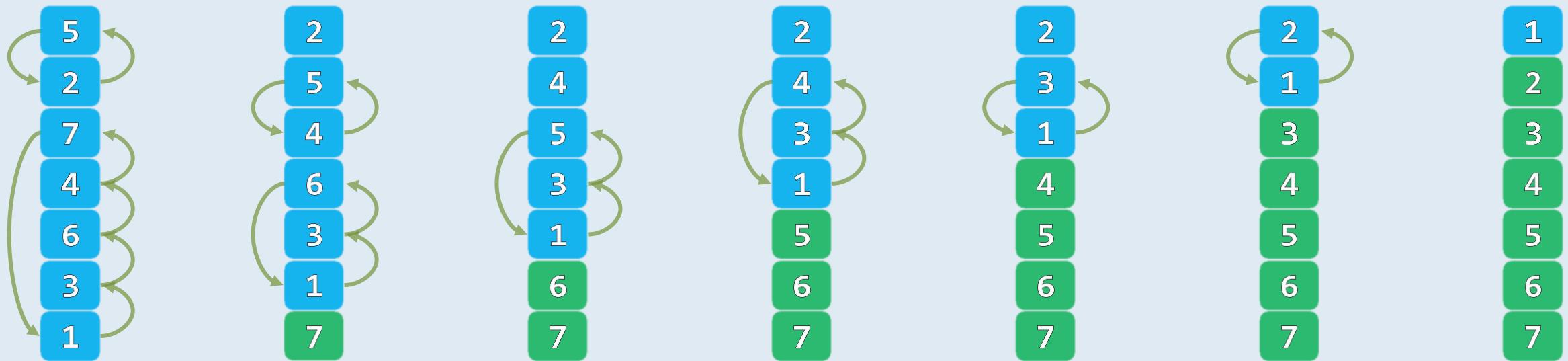
构思



- ❖ 问题：给定 n 个**可比较的元素**，将它们按（非降）序排列
- ❖ 观察：有序/无序序列中，任何/总有一对相邻元素顺序/逆序
- ❖ 扫描交换：依次比较每一对相邻元素；如有必要，交换之
- ❖ 若整趟扫描都没有进行交换，则排序完成；否则，再做一趟扫描交换



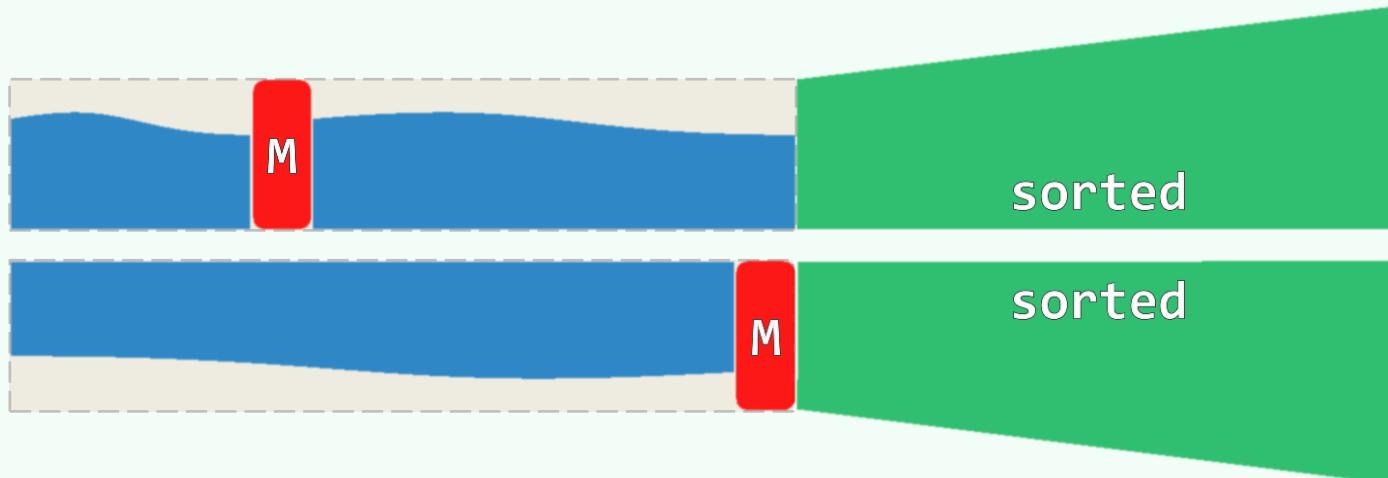
基本版



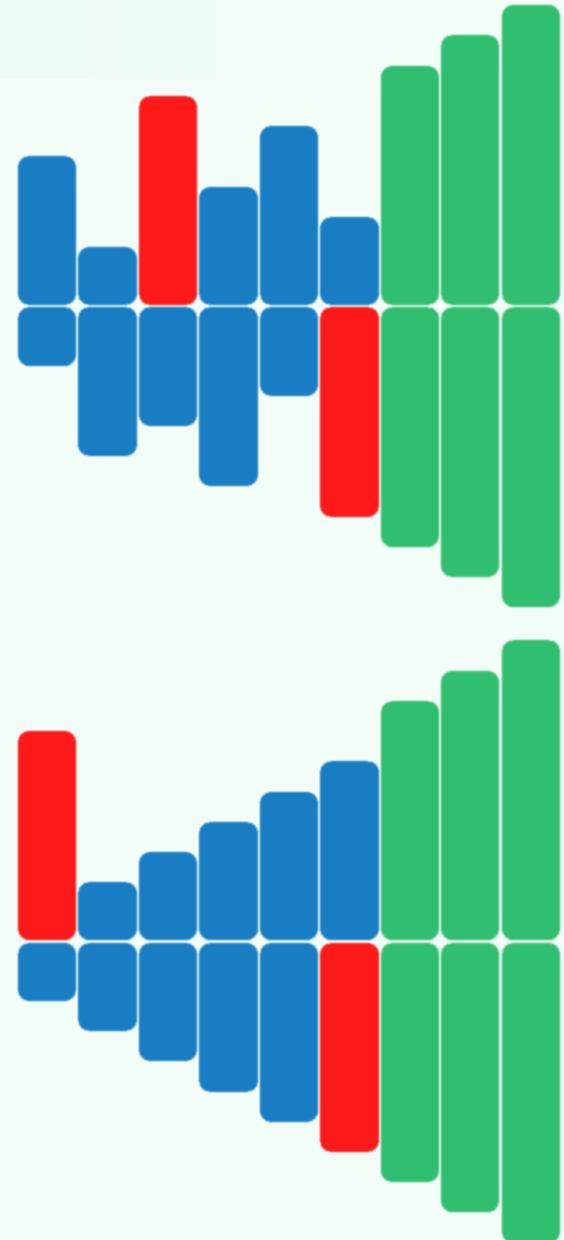
```
❖ template <typename T> Vector<T>::bubbleSort( Rank lo, Rank hi ) {  
    while( lo < --hi ) //逐趟起泡扫描 ( 输入保证 : 0 <= lo < hi <= size )  
        for( Rank i = lo; i < hi; i++ ) //若相邻元素  
            if( _elem[i] > _elem[i + 1] ) //逆序  
                swap( _elem[i], _elem[i + 1] ); //则交换  
}
```

正确性

- ❖ Loop Invariant : 经 k 趟扫描交换后，最大的 k 个元素必然就位
- ❖ Convergence : 经 k 趟扫描交换后，问题规模缩减至 $n-k$
- ❖ Correctness : 经至多 n 趟扫描后，算法必然终止，且能给出正确解答

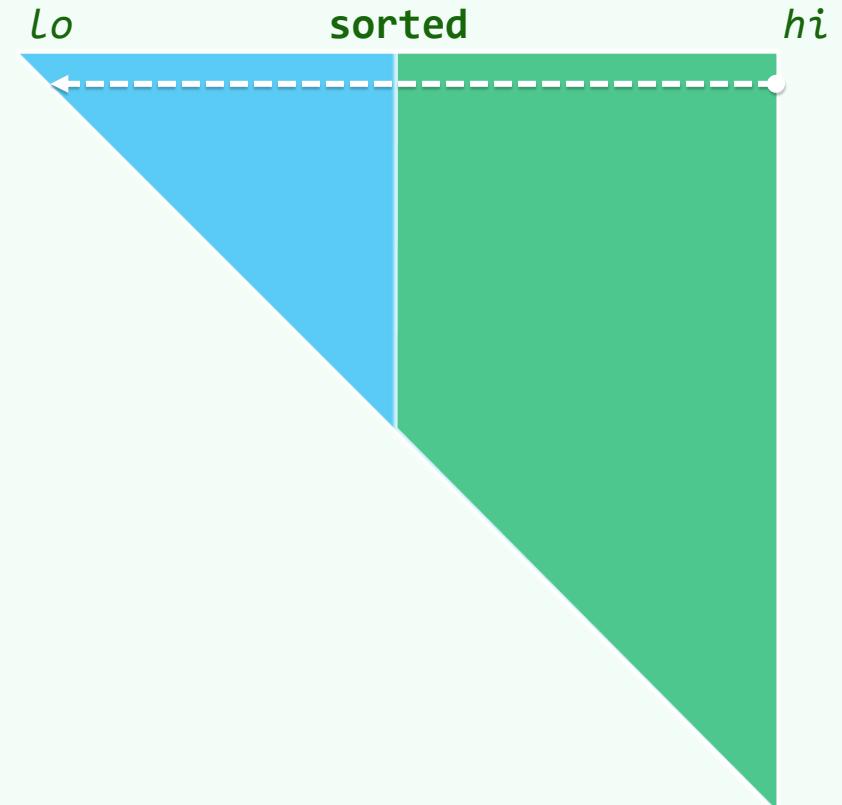


- ❖ $n-1$ 趟起泡扫描一定足够，但往往不必，比如...
- ❖ $[hi]$ 就位后， $[lo, hi)$ 可能已经有序 (sorted) —— 此时，应该可以...



提前终止版

```
❖ template <typename T> void Vector<T>::bubbleSort( Rank lo, Rank hi ) {  
    for( bool sorted = false; sorted = !sorted; )  
        for( Rank i = lo; i < hi - 1; i++ )  
            if( _elem[i] > _elem[i + 1] ) {  
                swap( _elem[i], _elem[i + 1] );  
                sorted = false; //仍未完全有序  
            } //else ... 提前终止  
    }  
}
```



❖ 有改进，但仍有继续改进的余地，比如...

❖ [hi]就位后，尽管[lo,hi)未必有序，但某后缀[last,hi)可能有序——此时，应该可以...

跳跃版

❖ template <typename T> void Vector<T>::bubbleSort(Rank lo, Rank hi) {

for(Rank last = --hi; lo < hi; hi = last)

for(Rank i = last = lo; i < hi; i++)

if(_elem[i] > _elem[i + 1]) {

swap(_elem[i], _elem[i + 1]);

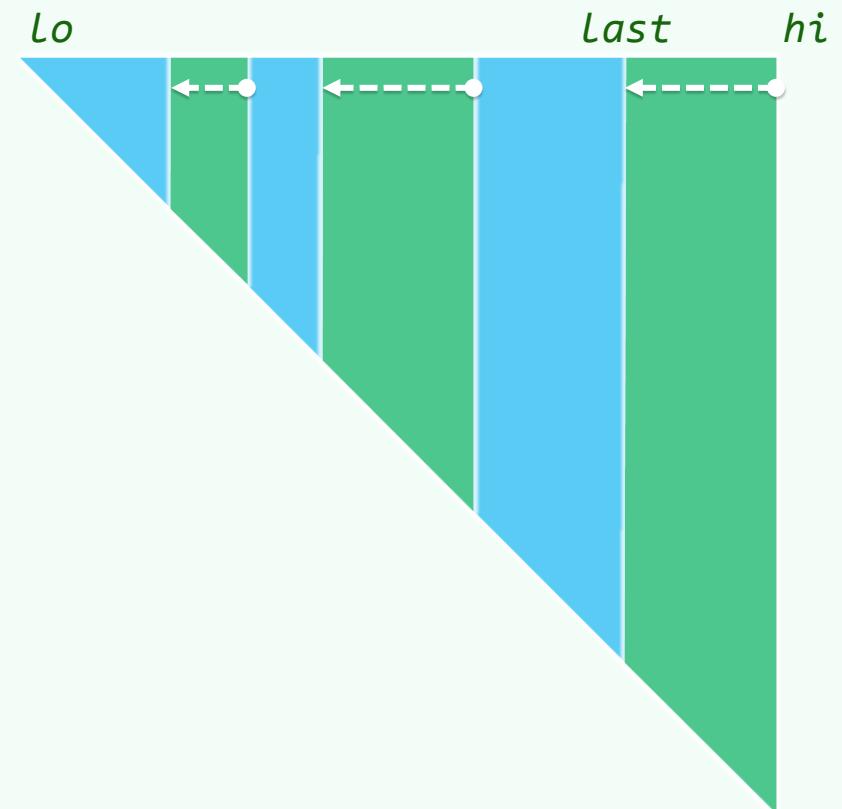
last = i; //逆序对只可能残留于[lo, last)

 }

}

❖ A[lo, last) <= A[last, hi)

A[lo, last] < A(last, hi)



综合评价

❖ 时间效率：最好 $\theta(n)$ ，最坏 $\theta(n^2)$

❖ 输入含重复元素时，算法的稳定性（stability）是更为细致的要求

重复元素在输入、输出序列中的相对次序，是否保持不变？

- 输入： 6, 7a, 3, 2, 7b, 1, 5, 8, 7c, 4

- 输出： 1, 2, 3, 4, 5, 6, 7a, 7b, 7c, 8 //stable

1, 2, 3, 4, 5, 6, 7a, 7c, 7b, 8 //unstable

❖ 以上起泡排序算法是稳定的吗？

是的！毕竟在起泡排序中，唯有相邻元素才可交换

❖ 在if一句的判断条件中，若把">"换成">="，将有何变化？