

14-A1

排序

快速排序：轴点

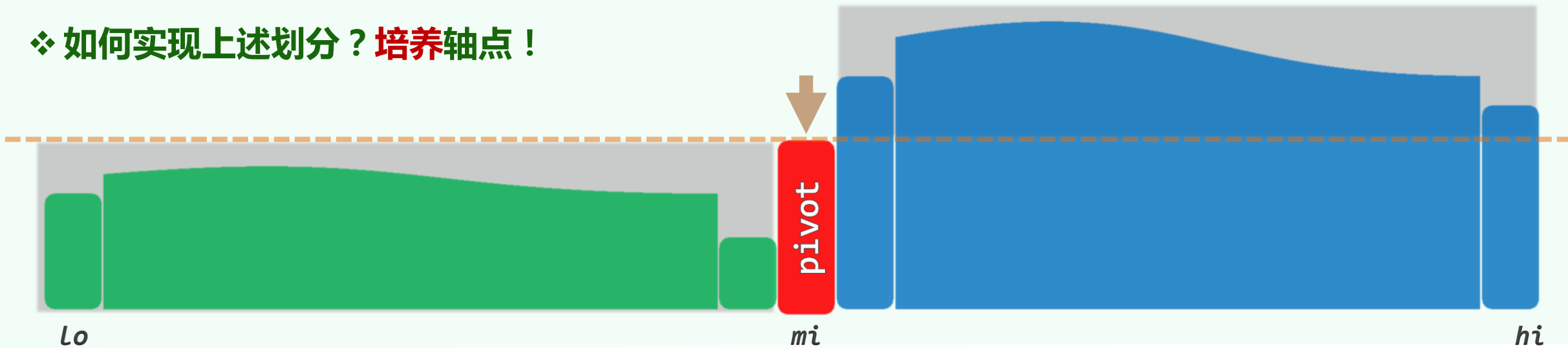
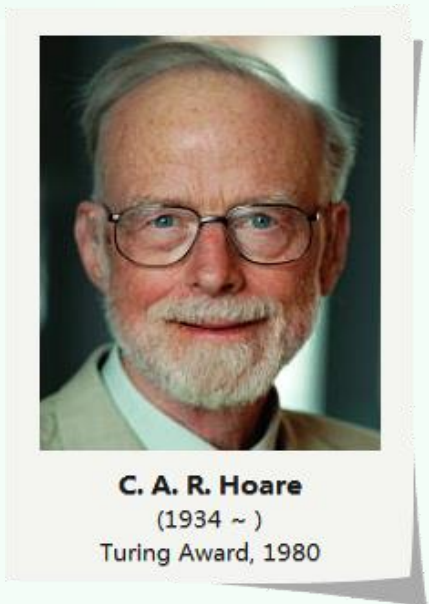
邓俊辉

deng@tsinghua.edu.cn

左朱雀之茕茕兮，右苍龙之蹏蹏

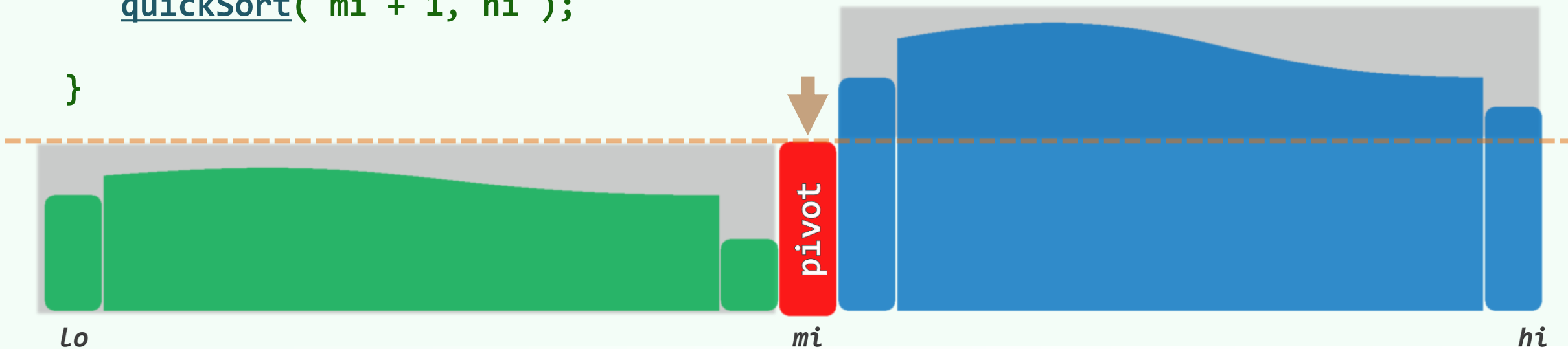
分而治之

- ❖ pivot : 左侧/右侧的元素 , 均不比它更大/更小
- ❖ 以轴点为界 , 自然**划分** : $\max([0, mi)) \leq \min(mi, hi)$
- ❖ 前缀、后缀各自 (递归) 排序之后 , 原序列自然有序
$$\text{sorted}(S) = \text{sorted}(S_L) + \text{sorted}(S_R)$$
- ❖ mergesort的难点在于**合** , 而quicksort在于**分**
- ❖ 如何实现上述划分 ? **培养**轴点 !



快速排序

```
❖ template <typename T> void Vector<T>::quickSort( Rank lo, Rank hi ) {  
    if ( hi - lo < 2 ) return;  
  
    Rank mi = partition( lo, hi ); //能否足够高效?  
  
    quickSort( lo, mi );  
  
    quickSort( mi + 1, hi );  
  
}
```



轴点

- ❖ 坏消息：在原始序列中，轴点**未必**存在...
- ❖ 必要条件：轴点必定已然**就位** // 尽管反之不然
- ❖ 特别地：在有序序列中，所有元素**皆为**轴点
反之亦然
- ❖ 快速排序：就是将所有元素**逐个转换**为轴点的过程
- ❖ **derangement**：任何元素都不在原位
比如，顺序序列循环移位
- ❖ 好消息：不需要很多次**交换**，即可使**任一**元素转换为轴点
- ❖ 问题：如何交换？成本多高？

