

- 教材讨论
 - DH第6章第129-139页
 - DH第7章第159-170页

问题1： 算法的效率

- linear search的时间复杂度是 $O(N)$
binary search的时间复杂度是 $O(\log N)$
请综合你对DH第129-139页的理解，谈谈你是如何理解这两句话的

问题1： 算法的效率

- linear search的时间复杂度是 $O(N)$
binary search的时间复杂度是 $O(\log N)$
请综合你对DH第129-139页的理解，谈谈你是如何理解这两句话的
- 关键点
 - 对于不同的输入，时间相同吗？
 - 计时的单位是什么？
 - big-O是什么意思？

问题1: 算法的效率 (续)

- 想想看, 有没有时间复杂度为 $O(1)$ 的search?

问题1： 算法的效率 (续)

- 如何理解big-O的鲁棒性？
（分别有什么优缺点？）

问题1: 算法的效率 (续)

- 你会分析insertion sort的时间复杂度吗?

5

6 3 1 8 7 2 4

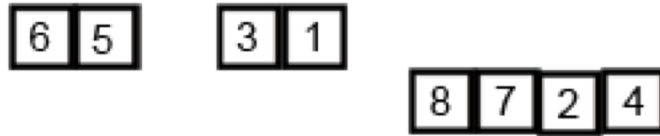
问题1: 算法的效率 (续)

- 你会分析insertion sort的时间复杂度吗?

```
i ← 1
while i < length(A)
  j ← i
  while j > 0 and A[j-1] > A[j]
    swap A[j] and A[j-1]
    j ← j - 1
  end while
  i ← i + 1
end while
```

问题1: 算法的效率 (续)

- 你会分析merge sort的时间复杂度吗?



问题1: 算法的效率 (续)

- 你会分析merge sort的时间复杂度吗?

```
function merge_sort(list m)
  // Base case. A list of zero or one elements is sorted, by definition.
  if length of m ≤ 1 then
    return m

  // Recursive case. First, divide the list into equal-sized sublists
  // consisting of the first half and second half of the list.
  // This assumes lists start at index 0.
  var left := empty list
  var right := empty list
  for each x with index i in m do
    if i < (length of m)/2 then
      add x to left
    else
      add x to right

  // Recursively sort both sublists.
  left := merge_sort(left)
  right := merge_sort(right)

  // Then merge the now-sorted sublists.
  return merge(left, right)
```

问题1: 算法的效率 (续)

- 你会分析merge sort的时间复杂度吗?

```
function merge(left, right)
  var result := empty list

  while left is not empty and right is not empty do
    if first(left) ≤ first(right) then
      append first(left) to result
      left := rest(left)
    else
      append first(right) to result
      right := rest(right)

  // Either left or right may have elements left; consume them.
  // (Only one of the following loops will actually be entered.)
  while left is not empty do
    append first(left) to result
    left := rest(left)
  while right is not empty do
    append first(right) to result
    right := rest(right)
  return result
```

问题2：问题的难度

- reasonable和tractable都表述多项式时间，区别是什么？

问题2： 问题的难度 (续)

- “算法的效率和问题的难度互为上下界”
结合之前的sort，谈谈你对这句话的理解

问题2： 问题的难度 (续)

- 对于intractable problem，怎么办？