Quicksort

Julián García

Sorting

<u>Input:</u>

A sequence of *n* elements

$$\langle a_1, a_2, ..., a_n \rangle$$

<u>Output:</u>

A permutation of the input sequence

$$\langle a'_1, a'_2, ..., a'_n \rangle$$

do it efficiently.

such that $a'_1 \leq a'_2 \leq ... \leq a'_n$

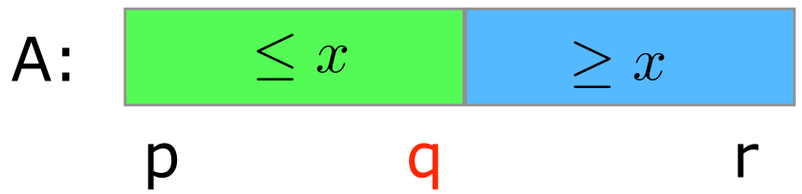
Quicksort

- Divide and conquer algorithm.
- Recursive
- Sorts in place (no extra arrays!).
- Popular: Top-10 algorithms 20th century (SIAM).

Quicksort strategy

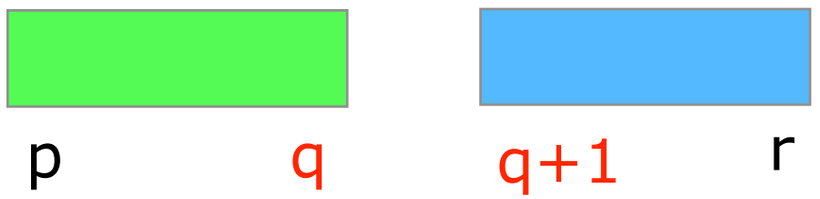
Input:

Divide:

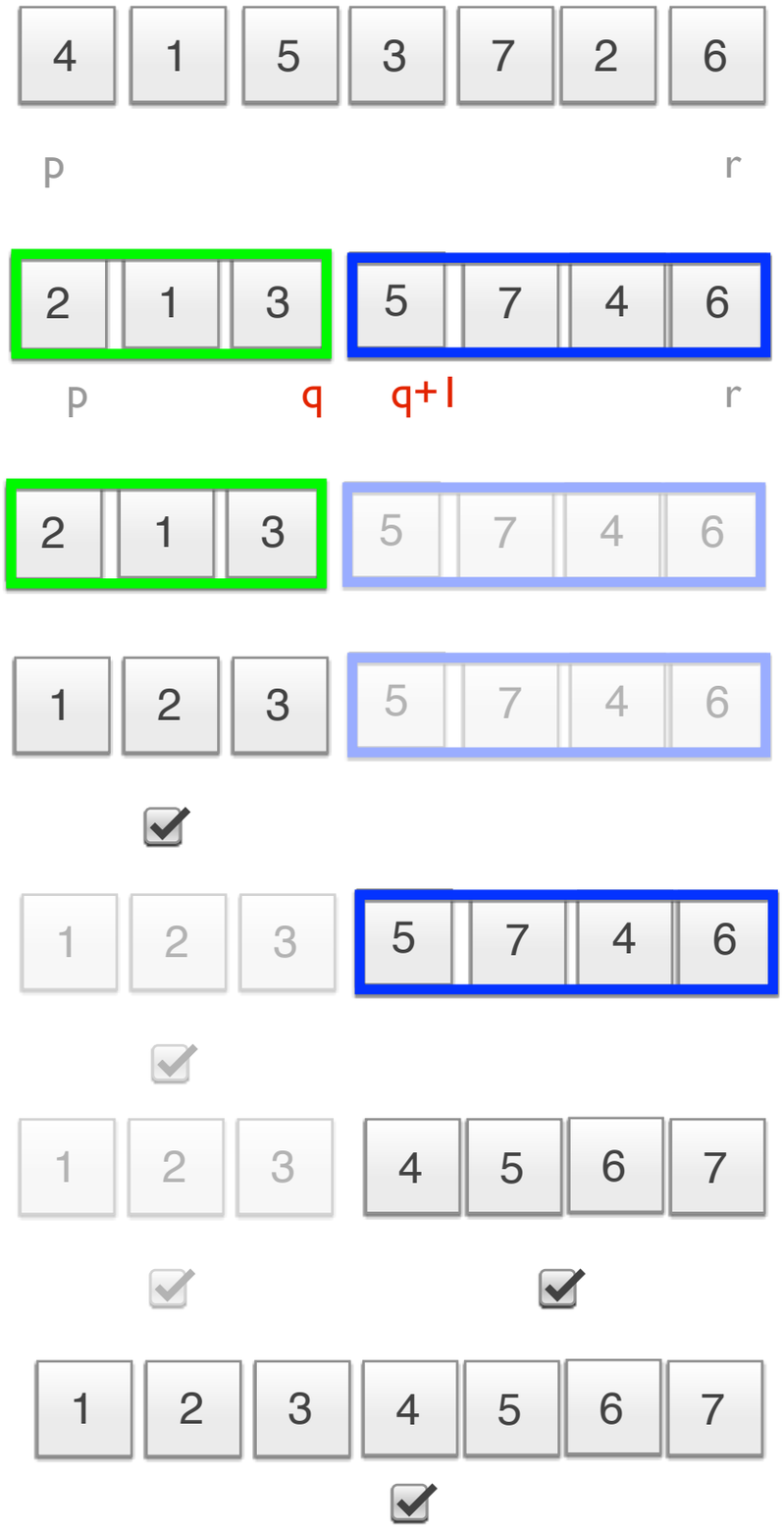


(partition step)

Conquer: Recursively solve two smaller problems.



Combine: In-place! so we are done.



(partition step)

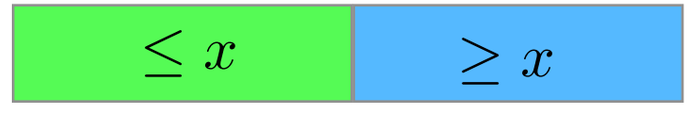
sort left

sort right

done!

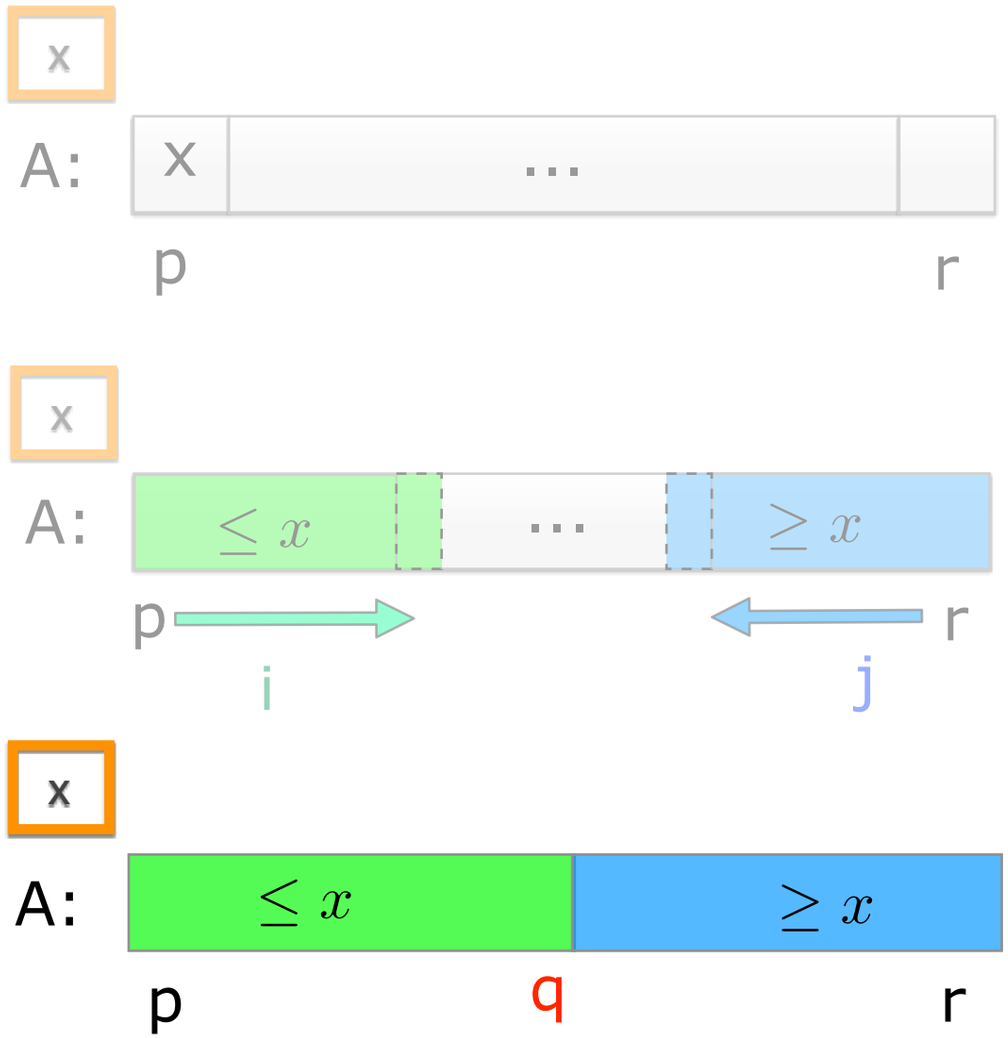
Is it non-trivial?
(exit door)

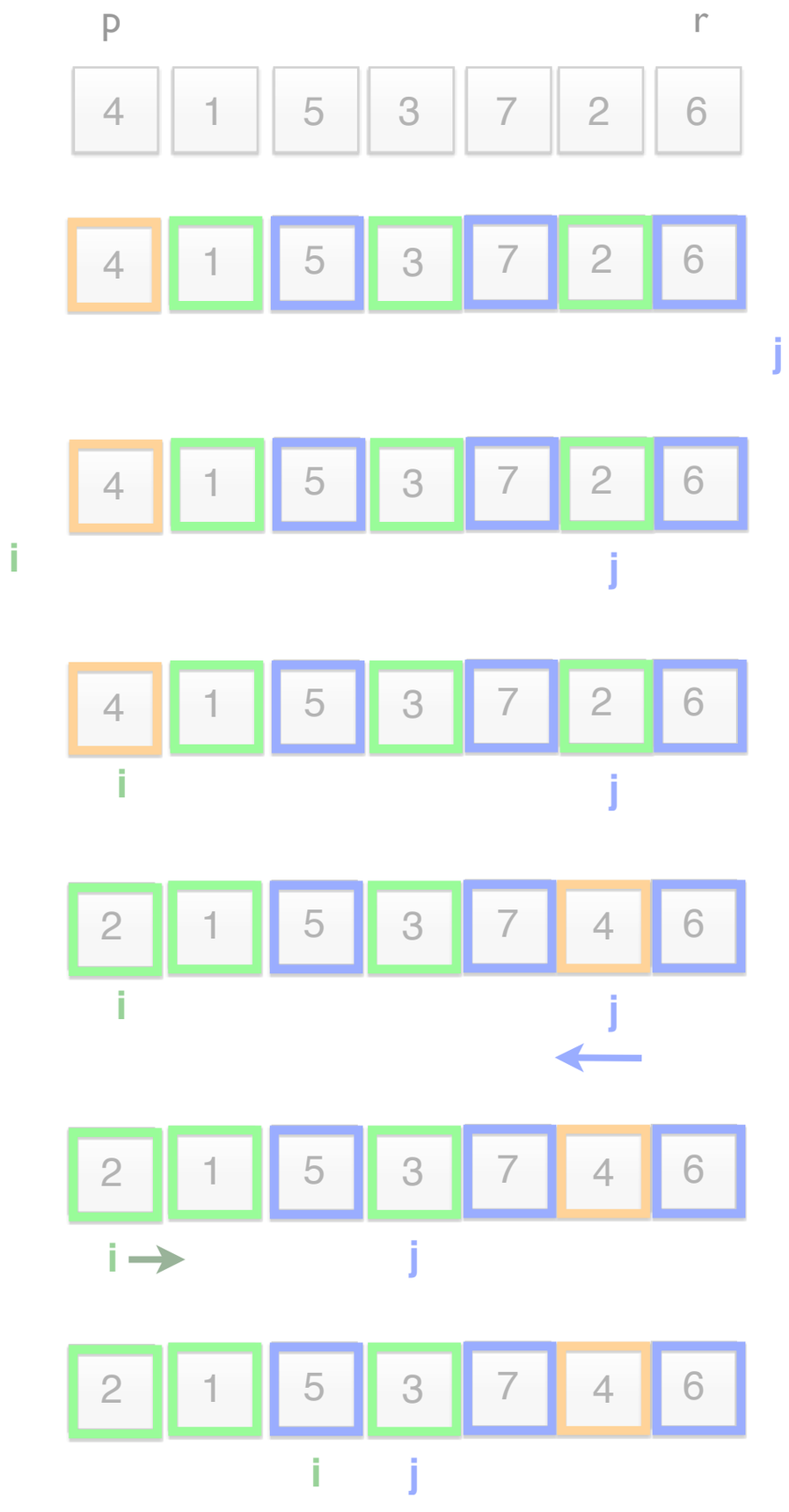
```
QUICKSORT(A, p, r)
1   if p < r
2       q = PARTITION(A, p, r)
3       QUICKSORT(A, p, q)
4       QUICKSORT(A, q + 1, r)
```

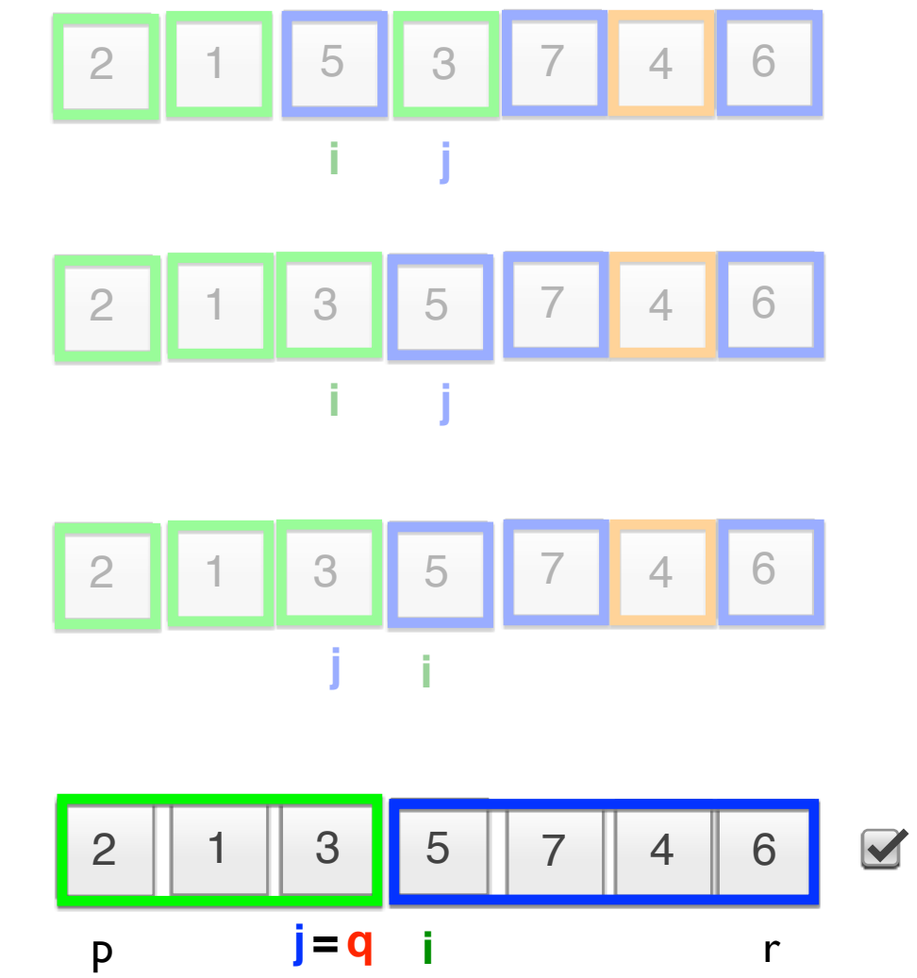


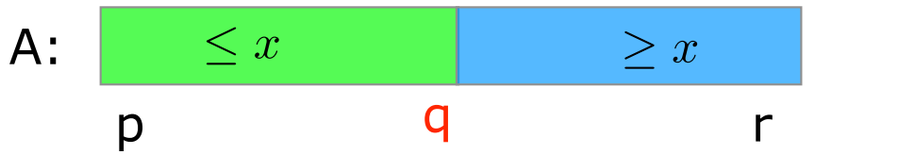
partition really does all the work!

PARTITION(A, p, r)

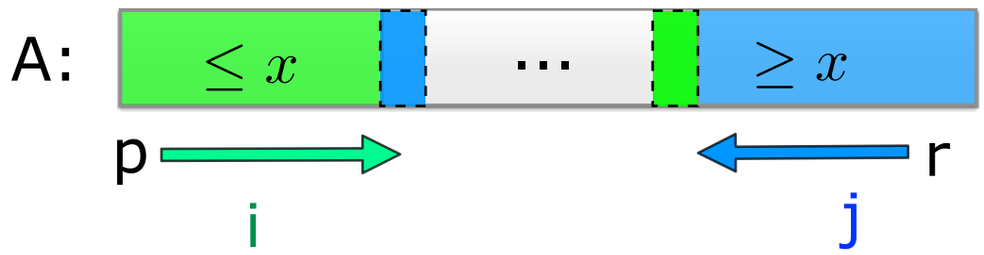


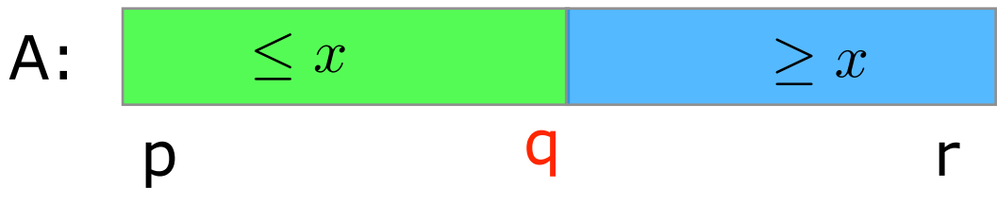

p
r
4 1 5 3 7 2 6
4 1 5 3 7 2 6
j
4 1 5 3 7 2 6
i
j
4 1 5 3 7 2 6
i
j
2 1 5 3 7 4 6
i
j
2 1 5 3 7 4 6
i
j
2 1 5 3 7 4 6
i
j


2 1 5 3 7 4 6
i
j
2 1 3 5 7 4 6
i
j
2 1 3 5 7 4 6
j
i
2 1 3 5 7 4 6
p
j=q
i
r


A:
$\leq x$
$\geq x$
p
q
r

```
PARTITION(A, p, r)
1   x = A[p]
2   i = p − 1
3   j = r + 1
4   while TRUE
5       repeat
6           j = j − 1
7       until A[j] ≤ x
8       repeat
9           i = i + 1
10      until A[j] ≥ x
11      if i < j
12          exchange A[i] with A[j]
13      else return j
```





Key points

- Several ways to do the partition (check lecture handout, experiment)

- Number of comparisons depends on the choice of the pivot, i.e., data.
  - Best case: we partition in halves all along
  - Worst case: ordered list (partition sizes: 1, n-1)

- The best case is to be expected.
  On average quicksort is very efficient.

Summing up

- Divide and conquer
    - Divide: Partition *in-place*
        - Conquer: Order each partition recursively
- Partition algorithms vary, and affect the process dramatically
- Efficient on average

Thank you.

More info: lecture handout

Literature:

- Leiserson, Charles E., Ronald L. Rivest, and Clifford Stein. ***Introduction to algorithms*****.** Edited by Thomas H. Cormen. The MIT press, 2001.
- Sedgewick, Robert. and Wayne, Kevin. ***Algorithms*****.** Pearson Education, 2011.

garcia@evolbio.mpg.de

http://garciajulian.com