Sorting

Eugeniy E. Mikhailov

The College of William & Mary

Lecture 07

Bubble sort method

Some one give us a vector of unsorted numbers.
We want to obtain the vector sorted in ascending order.

- assign `IndexOfTheLastToCheck` be the *index* of the vector end

1. start sweeping from the beginning of the vector
2. Compare the 2 consequent elements till we reach the `IndexOfTheLastToCheck`
3. if the left element is larger we swap these 2 elements
4. move to the next pair to the right i.e. move to the item 2
    - notice that at the end of the sweep the *index* of the last element to check holds the largest element
    - so next sweep does not have to be that long.
    - it is shorter by one element
    - i.e. the *index* of the last element to check should be decreased by 1
5. decrease `IndexOfTheLastToCheck` by 1
6. if `IndexOfTheLastToCheck` > 1 repeat from the item 1

$x = [3, 1, 4, 5, 2]$

first sweep

$x = [\widehat{3, 1}, 4, 5, 2]$ swap

$x = [1, 3, 4, 5, 2]$ after swap

$x = [1, \widehat{3, 4}, 5, 2]$ no swap

$x = [1, 3, \widehat{4, 5}, 2]$ no swap

$x = [1, 3, 4, \widehat{5, 2}]$ swap

$x = [1, 3, 4, 2, 5]$ sweep done

new sweep

$x = [\widehat{1, 3}, 4, 2, 5]$ no swap

$x = [1, \widehat{3, 4}, 2, 5]$ no swap

$x = [1, 3, \widehat{4, 2}, 5]$ swap

$x = [1, 3, 2, 4, 5]$ sweep done

new sweep

$x = [\widehat{1, 3}, 2, 4, 5]$ no swap

$x = [1, \widehat{3, 2}, 4, 5]$ swap

$x = [1, 2, 3, 4, 5]$ sweep done

last sweep

$x = [\widehat{1, 2}, 3, 4, 5]$ no sweep

$x = [1, 2, 3, 4, 5]$ sweep done

Bubble sort properties

- This is the worst of all working algorithm!
- The execution time of this algorithm is $\mathcal{O}(N^2)$
- Never use it in the real life!
- However it is very simple to program, and does not require extra memory for execution.

Quick sort method

Much better yet simple algorithm
Let's discuss recursive realization
We will name our sorting function as `qsort`.

- choose a pivot point value
  - let's choose the pivot at the middle of the vector
  - pivotIndex=floor(N/2)
  - pivotValue=x(pivotIndex)
- create two vectors which hold lesser and larger than pivotValue elements of the input vector.
- now concatenate the result of
  xs=[`qsort`(lesser), pivotValue, `qsort`(larger)]
- done

Quick sort summary

- usually fast
- typical execution time $\mathcal{O}(N \log_2 N)$
- but it is not guaranteed
  - However for certain input vectors execution time could be as long as $\mathcal{O}(N^2)$