Data Structures and Algorithms

Lecture 07

Aniket Basu Roy

BITS Pilani Goa Campus

2026-01-27 Wed

Agenda

Quicksort

- Description
- Proof of Correctness
- Time Complexity

Quicksort

Input: $A[p:r]$

Divide

Conquer

Combine

Output: $A[p:r]$ is sorted.

Quicksort

Input: $A[p:r]$

Divide

Partition $A[p:r]$ into two (possibly empty) subarrays, $A[p:q-1]$ and $A[q+1:r]$, such that $\forall a \in A[p:q-1]$ and $\forall b \in A[q+1:r]$, $a \leq A[q] \leq b$.

Conquer

Recursively call `QUICKSORT` on the two subarrays, $A[p:q-1]$ and $A[q+1:r]$.

Combine

Nothing to combine.

Output: $A[p:r]$ is sorted.

Quicksort

```
QUICKSORT(A, p, r)
if p < r
    // Partition the subarray around the pivot, which ends up in A[q].
    q = PARTITION(A, p, r)
    QUICKSORT(A, p, q − 1)  // recursively sort the low side
    QUICKSORT(A, q + 1, r)  // recursively sort the high side
```

Merge Sort (Recall)

```
MERGE-SORT(A, p, r)
if p ≥ r                              // zero or one element?
    return
q = ⌊(p + r)/2⌋                       // midpoint of A[p : r]
MERGE-SORT(A, p, q)                   // recursively sort A[p : q]
MERGE-SORT(A, q + 1, r)               // recursively sort A[q + 1 : r]
// Merge A[p : q] and A[q + 1 : r] into A[p : r].
MERGE(A, p, q, r)
```

Quicksort

```
QUICKSORT(A, p, r)
if p < r
    // Partition the subarray around the pivot, which ends up in A[q].
    q = PARTITION(A, p, r)
    QUICKSORT(A, p, q − 1)  // recursively sort the low side
    QUICKSORT(A, q + 1, r)  // recursively sort the high side
```

Quicksort

```
PARTITION(A, p, r)
x = A[r]                              // the pivot
i = p − 1                             // highest index into the low side
for j = p to r − 1                    // process each element other than the pivot
    if A[j] ≤ x                       // does this element belong on the low side?
        i = i + 1                     // index of a new slot in the low side
        exchange A[i] with A[j]       // put this element there
exchange A[i + 1] with A[r]           // pivot goes just to the right of the low side
return i + 1                          // new index of the pivot
```

Quicksort

Proof of Correctness

Quicksort

Proof of Correctness

- Correctness of the `PARTITION` function
  - Loop Invariant
- Proof by Induction on the number of array elements

Quicksort

Proof of Correctness

- Correctness of the `PARTITION` function
  - Loop Invariant

Quicksort

Proof of Correctness

- Correctness of the `PARTITION` function
  - Loop Invariant
    - Initialization
    - Maintenance
    - Termination

Quicksort

Time Complexity

Quicksort

Time Complexity

- (Un)balanced Partitions
  - Worst-case
  - Best-case
  - Balanced

Quicksort

Worst-case Time Complexity

$$T(n) = \max_{0 \leq q \leq n-1} \big(T(q) + T(n - q - 1) \big) + \Theta(n)$$