



Tutorial 7: Advanced Sorting Algorithms

CAB301 - Algorithms and Complexity

School of Computer Science, Faculty of Science

Agenda

1. **Lecture Recap:** Advanced Sorting Algorithms

- Merge Sort
- Quick Sort
- Heap Sort

2. **Tutorial Questions + Q&A**

Merge Sort

<div style="display: flex"> <div style="flex: 0.5">

Divide and Conquer algorithm, relies on a **merge** operation:

- How to combine two sorted arrays into a single sorted array?

ALGORITHM *MergeSort*($A[i..j]$)

 if $i < j$

$m \leftarrow \lfloor (i + j) / 2 \rfloor$

MergeSort($A[i..m]$)

MergeSort($A[m + 1..j]$)

Merge($A[i..j], m$)

</div> <div style="flex: 0.5; width: 500px">

Quick Sort

<div style="display: flex"> <div style="flex: 0.5">

Divide and Conquer algorithm, relies on a **partition** operation that, given a pivot, divides the array into two parts:

- Left part contains elements less than the pivot, and right part greater.

ALGORITHM *QuickSort*($A[l..r]$)
 if $l < r$
 $s \leftarrow \text{Partition}(A[l..r])$
 QuickSort($A[l..s - 1]$)
 QuickSort($A[s + 1..r]$)

</div> <div style="flex: 0.5; display: flex; flex-direction: column; justify-content: center;

Heap Sort

<div style="display: flex"> <div style="flex: 0.5">

Heap Sort keeps the array as a **max-heap**:

- Complete binary tree
- Each node is no less than its children.

Repeatedly perform **Maximum Key Deletion**:

1. Exchange the root's key with the last key.
2. Decrease the heap size by 1.
3. **Heapify** the complete binary tree.

</div> <div style="flex: 0.4; display: flex; flex-direction: column; justify-content: center;