DIVIDE AND CONQUER

CSX3009 ALGORITHM DESIGN

KWANKAMOL NONGPONG, PH.D.

DIFFERENT WAYS TO DESIGN ALGORITHMS

- Incremental approach
  - Insertion sort
- Divide-and-conquer approach

CONCEPTS

- Partitions the problem into independent subproblems
- Solves the subproblem recursively
- Then, combines their solutions to solve the original problem.
- Advantage:
  - Running times are often easily determined

STEPS

- Divide the problem into a number of subproblems.
- Conquer the subproblems by solving them recursively.
- Combine the solutions to the subproblems into the solution for the original problem.

MERGE SORT

- Divide
  - Divide the n-element sequence to be sorted into two subsequences of n/2 elements each
- Conquer
  - Sort the two subsequences recursively using merge sort.
- Combine
  - Merge the two sorted subsequences to produce the sorted answer.

MERGE SORT PSEUDOCODE

```
Merge-Sort(A, p, r)
    if p < r then
        q ← ⌊ (p+r) / 2 ⌋
        Merge-Sort(A, p, q)
        Merge-Sort(A, q+1, r)
        Merge(A, p, q, r)
```

EXAMPLE

- A = { 5, 2, 4, 6, 1, 3, 2, 6 }

ANALYZING DIVIDE-AND-CONQUER ALGORITHMS

- Recurrence equation describes the overall running time on a problem of size n in terms of the running time on smaller inputs.

- $$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq c \\ aT\left(\frac{n}{b}\right) + D(n) + C(n) & \text{otherwise} \end{cases}$$

ANALYSIS OF MERGE SORT

- Divide
  - Divide the n-element sequence to be sorted into two subsequences of n/2 elements each.
  - Computes the middle of the subarray, takes constant time.
    - $D(n) = \Theta(1)$
- Conquer
  - Sort the two subsequences recursively using merge sort.
  - Recursively solve two subproblems, each of size n/2
    - $2T(n/2)$
- Combine
  - Merge the two sorted subsequences to produce the sorted answer.
  - Merge procedure on an n-element subarray takes $\Theta(1)$
    - $C(n) = \Theta(n)$

WORST-CASE RUNNING TIME: MERGE SORT

- $T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 2T\left(\frac{n}{2}\right) + \Theta(n) & \text{if } n > 1 \end{cases}$

COMPARISON

- Insertion sort implemented in machine language for supercomputer
- Merge sort implemented in a high-level language on a PC
- Which one is faster?
  - $\frac{2((10)^6)^2 \text{ instructions}}{10^8 \text{ instructions/second}}$ = 20,000 seconds ≈ 5.56 hours
  - $\frac{5\,(10)^6 \lg 10^6 \text{ instructions}}{10^6 \text{ instructions/second}}$ = 1,000 seconds ≈ 16.67 minutes