University of Colombo School of Computing
SCS 1304 - Problem Solving Strategies and Computation Approaches

Lab Sheet 08

Brute Force & Divide-and-Conquer

Answer the following questions and you are allowed to use non programmable calculators.

A. A logistics company is sorting delivery addresses from various cities. They have divided the addresses into several regional subarrays. To optimize the sorting process, they use the Merge Sort algorithm.

1. Fill out the following trace table to show the steps taken by Merge Sort when sorting the array **[15, 42, 7, 8, 35, 22, 3, 10]**. Include all intermediate steps for both splitting and merging.
2. Draw the Split-Merge tree for the array **[15, 42, 7, 8, 35, 22, 3, 10]**. Represent the divisions at each step and the merging process leading to the final sorted array.
3. Given that the array has 8 elements, calculate the total number of comparisons made during the merge steps in Merge Sort. Explain how this relates to the time complexity of Merge Sort.

B. A retail company is analyzing daily sales data to identify the period with the highest sales. The sales figures for 6 consecutive days are:

[200, -150, 300, -250, 500, -100].

1. Explain the Brute Force Approach to find the maximum subarray sum in the sales data **[200, -150, 300, -250, 500, -100].** Describe the steps involved and the computational complexity.
2. Calculate the maximum subarray sum using the Brute Force method. List all possible subarrays and identify the one with the highest sum. Provide the total number of comparisons made during the process.
3. Briefly explain why the Brute Force method might be inefficient for larger datasets and suggest a more efficient alternative algorithm.

C. A library system needs to quickly find books by their unique ID numbers. The book IDs are stored in a sorted array, and the system uses Binary Search to locate specific IDs.

1. Describe how the Binary Search algorithm works when trying to find a book with ID 457 in the following sorted array:

   [123, 234, 345, 456, 457, 567, 678, 789].

   Outline each step and explain why Binary Search is more efficient than a linear search in this case.

2. Calculate the number of comparisons Binary Search would make in the worst-case scenario to find any ID in an array of 256 sorted elements. Explain how the time complexity of Binary Search relates to the number of comparisons.

3. Suppose the library system occasionally deals with unsorted arrays due to data entry errors. Discuss how Binary Search would behave if applied to an unsorted array and recommend a strategy for handling such situations.

D. Use the **mathematical induction method** to fill in the blanks and prove an upper bound time complexity for a recursive function through the steps. Consider the following recursive function,

$$\mathbf{T(n) = T(n - 1) + n}, \text{ with } \mathbf{T(1) = 1}$$

Use mathematical induction to prove that: $T(n) = O(n^2)$
(That is, prove $T(n) \leq c \cdot n^2$ for some constant c and for all $n \geq 1$)

1. **Base Case**

   Check whether the inequality holds for n = 1.
   Try with c = 1:
   Is $T(1) \leq 1^2$?
   Write down your result:
   $T(1) =$ ________ $c \cdot 1^2 =$ ________

   Inequality holds? (Yes/No) → ________

2. **Inductive Hypothesis**

Assume the statement is true for n = k,

$T(k) \leq$ ______________

This is your assumption.

3. **Inductive Step**

Use the recurrence to write:

$\mathbf{T(k+1) =}$ ________________________

Now, substitute the assumption into it:

$\mathbf{T(k+1) \leq}$ ________________________

4. **Simplify & Compare**

Try to simplify the expression:

_________ $\mathbf{+ (k+1)}$

Then compare it to:

$\mathbf{c \cdot (k+1)^2 = c \cdot}$ ______________

$$\mathbf{c \cdot k^2 + (k+1) \leq c \cdot k^2 + 2c \cdot k + c}$$

$$\mathbf{(k+1) \leq 2c \cdot k + c}$$

Does the inequality hold? Try with **c = 1**, or increase if needed.

Inequality holds? (Yes/No) → ________

5. **Conclusion**

If both the base case and the inductive step hold, write your conclusion:

____

So, $\mathbf{T(n) = O(n^2)}$ for all $\mathbf{n \geq 1}$

Quiz

01. Which of the following best describes the Divide and Conquer approach?

A) Breaking a problem into smaller subproblems, solving each one independently, and combining the solutions.

B) Merging two sorted arrays into a single sorted array.

C) Iteratively improving a solution until optimal.

D) Sorting an array by comparing adjacent elements.

02. What is the primary advantage of using Divide and Conquer algorithms?

A) They always run in linear time.

B) They can efficiently solve problems by breaking them down into manageable parts.

C) They require less memory than other algorithms.

D) They are easier to implement than brute force methods.

03. Which algorithm is a classic example of the Divide and Conquer strategy?

A) Bubble Sort B) Quick Sort

C) Linear Search D) Selection Sort

04. In a Divide and Conquer algorithm, what is the 'Conquer' step responsible for?

A) Dividing the problem into smaller parts.

B) Solving each subproblem independently.

C) Combining the results of subproblems into a final solution.

D) Initializing the variables for the algorithm.

05. What is the time complexity of the Merge Sort algorithm?

A) O(n) B) O(n log n) C) O(n^2) D) O(log n)

06. Which of the following operations does NOT typically use a Divide and Conquer strategy?

A) Matrix multiplication B) Merge Sort

C) Insertion Sort D) Binary Search

07. In Merge Sort, what is the first step of the algorithm?

A) Merging two arrays

B) Sorting the array

C) Dividing the array into two halves

D) Swapping the elements

08. Which of the following is true about Merge Sort?

A) It is an in-place sorting algorithm.

B) It requires additional space proportional to the size of the array.

C) It works best on small datasets.

D) It is not stable.

09. What is the key operation performed during the 'Merge' step in Merge Sort?

A) Comparing and merging two sorted halves into a sorted array.

B) Swapping elements to place them in order.

C) Dividing the array into smaller subarrays.

D) Selecting the pivot element.

10. **Which of the following problems is typically solved using the Divide and Conquer approach?**

A) Fibonacci sequence

B) Matrix multiplication

C) Searching in an unsorted list

D) Traversing a linked list

11. **In the context of Divide and Conquer, what is 'Recurrence Relation' used for?**

A) To define the solution of a problem in terms of its subproblems.

B) To sort the subproblems.

C) To divide the problem into subproblems.

D) To combine the solutions of the subproblems.

12. **Which of the following statements is true about Merge Sort?**

A) It is a non-recursive algorithm.

B) It requires more memory than Quick Sort.

C) It is not suitable for large datasets.

D) It is less stable than Bubble Sort.

13. **What is the base case in a Divide and Conquer algorithm?**

A) The case where the problem cannot be divided further.

B) The case where the problem is divided into two parts.

C) The case where the problem is solved iteratively.

D) The case where the problem is merged with others.

14. Why is Merge Sort considered stable?

A) Because it doesn't swap equal elements.

B) Because it preserves the order of equal elements.

C) Because it works faster than other algorithms.

D) Because it uses less memory.

15. Which of the following is NOT a feature of the Divide and Conquer approach?

A) Parallelism B) Scalability

C) Complexity reduction D) Space efficiency

16. In which phase of the Divide and Conquer strategy is the problem size reduced?

A) Divide B) Conquer C) Combine D) Initialization

17. You are given a list of unsorted integers and you want to sort them using Merge Sort. If the list has 16 elements, how many times will the array be split during the Divide step?

A) 2 times B) 4 times C) 8 times D) 16 times

18. If you apply Merge Sort to an array with 256 elements, what is the approximate number of comparisons required?

A) 512 B) 2,048 C) 1,024 D) 4,096

19. You need to merge two sorted arrays of size 50 and 70, respectively, using the Merge Sort algorithm. How many comparisons will be made during the merge step?

A) 120 B) 60 C) 100 D) 150

20. **A company is using Divide and Conquer to split a large file into smaller chunks before processing. If the file is initially 64 GB, and each chunk is halved during each divide step, how many steps will it take to reach a chunk size of 1 GB?**

A) 6 steps B) 7 steps C) 8 steps D) 9 steps

21. **If you perform Merge Sort on a nearly sorted array of 1,000 elements, what will be the expected time complexity?**

A) O(n) B) O(n log n) C) O(n^2) D) O(log n)

22. **In a Divide and Conquer approach, if you split a task into 4 subproblems and combine them in O(n^2) time, what is the overall time complexity if each subproblem is solved in O(n)?**

A) O(n^3) B) O(n^2 log n) C) O(n log n) D) O(n^4)

23. **Using the Divide and Conquer strategy, if you split a problem into three parts, each of which takes O(n) time to solve, what is the overall time complexity?**

A) O(n^2) B) O(n log n) C) O(n) D) O(n^3)

24. **Suppose you have 8 elements in an array and you apply Merge Sort. After the first division, how many subarrays will be present?**

A) 2 B) 4 C) 8 D) 16