National University of Computer and Emerging Sciences, Lahore Campus



Course: Design and Analysis of Algorithms

Program: BS (CS, SE)

Due Date: Feb 22, 2024 (before 5 PM)

Sections: CS-4(G,H,K), SE-6A Exam: Assignment#1 Course Code: Semester: Total Marks: **CS-2009**Spring 2024

70

Instruction/Notes:

Instructions: Plagiarism in any form will not be tolerated. Only handwritten, hard copy assignments will be accepted.

Q#1: Derive the recurrence of following recursive algorithms. (Marks 12)

```
Part(a) Split Data (Arr[], left, right){
                                                           Part(b)
  If (left < right) {
                                                           Temp(A[], B[], N)
  Split Data (Arr, left, right-1)
                                                             if(N==1)
    Split Data (Arr, left, right)
                                                               return /
  ∠View Data (Arr, left right) N
                                                             Temp(A+N/2,B, N/2) + Temp(A, B + N/2, N/2)
                                                             Temp(A+N/2,B, \overline{N/2}) + Temp(A, B + N/2, N/2)
                                                             Temp(A+N/2,B, N/2) + Temp(A, B + N/2, N/2) =
                                                             Temp(A+N/2,B, N/2) + Temp(A, B + N/2, N/2)
View Data (Arr[], left, right){
                                                             Temp(A+N/2,B, N/2) + Temp(A, B + N/2, N/2)
  i=left
  N Arr[left+right] -
                                                             Temp(A+N/2,B, N/2) + Temp(A, B + N/2, N/2)
 While (i < right) {_____</pre>
                                                             Temp(A+N/2,B, N/2) + Temp(A, B + N/2, N/2)
    i = left_
                                                             Temp(A+N/2,B, N/2) + Temp(A, B + N/2, N/2)
    k = 1
                                                             Solve(A,B,N)
    while (j < right) {
      N Arr[k] = Arr[j]
                                                          Solve (A[], B[], N){
                                                             for(i=1 to N){
      print Arr[i]
      k = k+1
                                                               for(j=1 to i){
      j = j*2
                                                                  print A[i]*B[j]
                                                           Part(d): Stooge-Sort (Arr [], left, right){
Part (c):
                                                            if (Arr[left] > A[right])
Mystery (N){
                                                               exchange(A[left], A[right])
  If (N > 1)
    Print "Deriving recurrence is fun"
                                                             if (left > right)
    Mystery (2N/3) ___
                                                               return
    for (i=1 to N)
                                                             else{
      Print "Solving recurrence is fun"
                                                               third = (right - left + 1)/3
                                                               Stooge-Sort (Arr, left, right - third)
    Mystery (N/5)
                                                               Stooge-Sort (Arr, left + third, right)
                                                               Stooge-Sort (Arr, left, right - third)
}
                                                           Also provide the reason that how these recursive
                                                           calls will sort the array and why we need the third
                                                           recursive call with data ranging from Left to 3rd
                                                           quarter of the array.
```

FAST School of Computing

1918/7/64--1391

 $3T\left(\frac{2N}{3}\right) + O(1)$

Page 1

Q#2: Solving Recurrence (Marks: 50) 4 5 6 11213 71819

Use a recursion tree method to determine a good asymptotic upper bound on following recurrences. Please refer to the Appendix of your textbook for using arithmetic, geometric and harmonic series. Clearly show your working and don't forget to mention the relevant formula before writing the final answer.

Part(a): Linear Decay

- a) $T(N) = 3T(N-1) + \theta(1)$
- b) $T(N) = T(N-2) + O(N^2)$
- c) $T(N) = 2T(N-1) + \Theta(1)$
- d) $T(N) = T(N-1) + O(\frac{1}{N})$

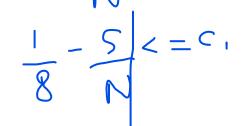
Part(b): Exponential decay with balanced split of data (Attempt any four questions for assignment from part b)

- e) $T(N) = 2T\left(\frac{N}{4}\right) + O(\log N)$
- f) $T(N) = 10T\left(\frac{N}{2}\right) + \Theta(1)$
- g) $T(N) = 2T\left(\frac{N}{2}\right) + O\left(\frac{N}{\log N}\right)$
- h) $T(N) = 4T\left(\frac{N}{2}\right) + O(N^2\sqrt{N})$
- i) $T(N) = 2T\left(\frac{N}{4}\right) + O(\sqrt{N})$
- $\mathbf{j}) \quad T(N) = 3T\left(\frac{N}{2}\right) + O(N^3)$
- k) $T(N) = 7T\left(\frac{N}{5}\right) + \Theta(1)$
- 1) $T(N) = 2T\left(\frac{N}{4}\right) + O(\sqrt{N})$
- m) $T(N) = 3T\left(\frac{2N}{3}\right) + O(1)$
- $\mathbf{n}) \ T(N) = 3T\left(\frac{4N}{5}\right) + \ \mathbf{O}(1)$

 $f(N) < = C_1 \times O_1(N)$ $\frac{1}{N} - 5N^2 < = C_1 \times O_1(N)$ $\frac{1}{N} - 5N^2 < = C_1 \times O_1(N)$

Part(c): Exponential decay with unbalanced split of data

- o) $T(N) = T\left(\frac{N}{4}\right) + T\left(\frac{3N}{4}\right) + O(N^2)$
- p) $T(N) = T\left(\frac{N}{10}\right) + T\left(\frac{9N}{10}\right) + O(N)$

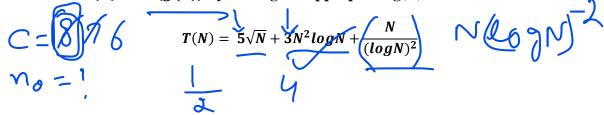


Q#3: Asymptotic Analysis (3 Marks)

a) Prove that $T(N) = \Theta(N^3)$ by finding appropriate constants.

$$T(N) = \frac{1}{8} N^3 - 5N^2$$

b) Prove that $T(N) = \Theta(g(N))$ by finding the appropriate g(N) and constants.



FAST School of Computing	Page
write down the recurrence of your pocudo code. (5 Marks)	
Q#4: Provide pseudo code of merge sort with in-place merge routine. write down the recurrence of your pseudo code. (5 Marks)	