CSC-654 Algorithms Analysis & Design
Homework 3

Mostafa Abdelmegeed

3/15/2023

1 Problem 4.1-5 from the textbook

Use the following ideas to develop a nonrecursive, linear-time algorithm for the maximum-subarray problem. Start at the left end of the array, and progress toward the right, keeping track of the maximum subarray seen so far. Knowing a maximum subarray of A[i..j], extend the answer to find a maximum subarray ending at index j+1 by using the following observation: a maximum subarray of A[1..j+1] is either a maximum subarray of A[1..j] or a subarray A[i..j+1], for some 1 ¡= i ¡= j+1. Determine a maximum subarray of the form A[i..j+1] in constant time based on knowing a maximum subarray ending at index j.

```
func maxSubArray(arr)
    max = -inf
    sum = 0
    start, end = 1
    for j = 1 to arr.length
        sum += arr[j]
        if arr[j] > sum
            sum = arr[j]
            start = j
        if sum > max
            max = sum
            end = j
    return arr[start:end]
```

2 Problem 4.2-2 from the textbook

Write pseudocode for Strassen's algorithm.

```
Assuming two matrices of same size and are padded with zeros to
maintain even-numbered dimension size
Given zeros(size), a method that returns a pre-filled matrix of
zeros with dimensions equal to size
func strassen(A, B)
    n = A.size
    C = zeros(A.size)
    if n == 1
```

```
    C[1][1] = A[1][1]*B[1][1]
    return C
n_sub = n/2
A11 = A[1:n_sub][1:n_sub]
A12 = A[1:n_sub][n_sub:n]
A21 = A[n_sub:n][1:n_sub]
A22 = A[n_sub:n][n_sub:n]
B11 = B[1:n_sub][1:n_sub]
B12 = B[1:n_sub][n_sub:n]
B21 = B[n_sub:n][1:n_sub]
B22 = B[n_sub:n][n_sub:n]
C11 = C[1:n_sub][1:n_sub]
C12 = C[1:n_sub][n_sub:n]
C21 = C[n_sub:n][1:n_sub]
C22 = C[n_sub:n][n_sub:n]
S1 = B12 - B22
S2 = A11 + A12
S3 = A21 + A22
S4 = B21 - B11
S5 = A11 + A22
S6 = B11 + B22
S7 = A12 - A22
S8 = B21 + B22
S9 = A11 - A21
S10 = B11 + B12
P1 = strassen(A11, S1)
P2 = strassen(S2, B22)
P3 = strassen(S3, B11)
P4 = strassen(A22, S4)
P5 = strassen(S5, S6)
P6 = strassen(S7, S8)
P7 = strassen(S9, S10)
C11 = P5 + P4 - P2 + P6
C12 = P1 + P2
C21 = P3 + P4
C22 = P5 + P1 - P3 - P7
C = [[C11, C1,2], [C21, C22]]
return C
```

3 Use the substitution method described in section 4.3 to prove that the solution to the recurrence $T(n) = T(\lfloor n/2 \rfloor) + 1$ is $\mathcal{O}(n \log n)$. Don't worry about the base case. (Also, as I said in class, be aware that the substitution method in section 4.3 is not the method of repeatedly expanding the recurrence until a pattern becomes clear, which this course does not cover. Apparently some other textbooks call that the "substitution method," but it is not a formal proof, and it is not what this question is asking for.)

Guess $T(n) = \mathcal{O}(n \log n)$

Goal $T(n) \leq cn \log n \quad c > 0,\ n \geq n_o$

Let $m = \lfloor n/2 \rfloor$

$T(\lfloor n/2 \rfloor) \leq c\lfloor n/2 \rfloor \log \lfloor n/2 \rfloor + 1$

$T(\lfloor n/2 \rfloor) \leq \frac{cn}{2}(\log n - \log 2) + 1$

$T(\lfloor n/2 \rfloor) = \frac{cn}{2}(\log n - 1) + 1$

$T(\lfloor n/2 \rfloor) = \frac{cn}{2} \log n - \frac{cn}{2} + 1$

$T(\lfloor n/2 \rfloor) \leq \frac{cn}{2} \log n \quad if\ c \geq 2$

$\therefore\ T(n) \subset \mathcal{O}(n \log n)$