Name: Alfredo Perez
COSC 40403 - Analysis of Algorithms: Homework 6
Due: 23:30 on October 21

1. (12 points total) Use the master method to give tight asymptotic bounds for the following recurrences. Show your work to receive partial credit.
   (a) (3 points) $T(n) = 2T(n/4) + 1$.
   (b) (3 points) $T(n) = 2T(n/4) + \sqrt{n}$.
   (c) (3 points) $T(n) = 2T(n/4) + n$.
   (d) (3 points) $T(n) = 2T(n/4) + n^2$.

Answer:

Using the Master Method:
a)
Compare $f(n)$ with $\log_a b$ where a = 2, b = 4, $f(n) = 1$.
$n^{\log_a b} = n^{\log_2 4} = \sqrt{n} > f(n) = 1$
By knowing this we can deduce we have a time complexity of $\Theta(\sqrt{n})$
b)
Let a = 2, b = 4
$\log_a b = \log_2 4 = n^2 = f(n) = n^2$ The time complexity for this recurrence is $\Theta(f(n)) \log_n$ $= \Theta(\sqrt{n} \log_n)$
c)
Let a = 2, b = 4
$\log_a b = \log_2 4 = \sqrt{n} < f(n) = n$ Thus, the time complexity is $\Theta(f(n)) = \Theta(n)$
d)
Let a = 2, b = 4
$\log_a b = \log_2 4 = \sqrt{n} < f(n) = n^2$.
Thus the time compelxity is of $\Theta(f(n)) = \Theta(n^2)$

2. (8 points) Use Strassen's algorithm to compute the matrix product $\begin{pmatrix} 2 & 3 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 11 & 13 \\ 17 & 19 \end{pmatrix}$. Show your work to receive partial credit.

Answer:

Strassen Algorithm

2) $\begin{pmatrix} 2 & 3 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 11 & 13 \\ 17 & 19 \end{pmatrix}$

$\begin{pmatrix} 2 & 3 \\ 5 & 7 \end{pmatrix} = \begin{pmatrix} A & A \\ A & A \end{pmatrix}$

$\begin{pmatrix} 11 & 13 \\ 17 & 19 \end{pmatrix} = \begin{pmatrix} B & B \\ B & B \end{pmatrix}$

$A_{1,1} = 2$, $A_{1,2} = 3$, $A_{2,1} = 5$, $A_{2,2} = 7$

$B_{1,1} = 11$, $B_{1,2} = 13$, $B_{2,1} = 17$, $B_{2,2} = 19$

$M_1 : (2+7)(11+19)$ $\quad 270$
$M_2 : (5+7)\,11$ $\quad 132$
$M_3 : (2)(13-19)$ $\quad -12$
$M_4 : 7(17-11)$ $\quad 42$
$M_5 : (2+3)\,19$ $\quad 95$
$M_6 : (5-2)(11+13)$ $\quad 72$
$M_7 : (3-7)(17+19)$ $\quad -144$

$C_{1,1} = 270 + 42 - 95 + -144$ $\quad 73$
$C_{1,2} = -12 + 95$ $\quad 83$
$C_{2,1} = 132 + 42$ $\quad 174$
$C_{2,2} = 270 - 132 + -12 + 72$ $\quad 198$

$= \begin{pmatrix} 73 & 83 \\ 174 & 198 \end{pmatrix}$

3. (10 points) Write pseudocode for Strassen's algorithm. Do not write your algorithm in Python (or any other programming language).

Answer:

Let A = $\begin{bmatrix} A^{11} & A^{12} \\ A^{21} & A^{22} \end{bmatrix}$ and B = $\begin{bmatrix} B^{11} & B^{12} \\ B^{21} & B^{22} \end{bmatrix}$

Strassen(A,B):

If n = 1 Return A x B

Else:

$M1 = Strassen(A^{11}, B^{12} - B^{22})$
$M2 = Strassen(A^{11} + A^{12}, B^{22})$
$M3 = Strassen(A^{21} + A^{22}, B^{11})$
$M4 = Strassen(A^{22}, B^{21} - B^{11})$
$M5 = Strassen(A^{11} + A^{22}, B^{11} + B^{22})$
$M6 = Strassen(A^{1}2 - A^{22}, B^{21} + B^{22})$
$M7 = Strassen(A^{11} - A^{21}, B^{11} + B^{12})$

$C^{11} = M_5 + M_4 - P_2 + P_6$
$C^{12} = M_1 + P_2$
$C^{21} = M_3 + M_4$
$C^{22} = M_1 + M_5 - M_3 - M_7$

Return C

4. (20 points total) The version of PARTITION given in lecture is not the original partitioning algorithm. Here is the original partition algorithm, which is due to C. A. R. Hoare in 1962.

```
HOARE-PARTITION(A, p, r)
1   x = A[p]
2   i = p − 1
3   j = r + 1
4   while TRUE
5       repeat
6           j = j − 1
7       until A[j] ≤ x
8       repeat
9           i = i + 1
10      until A[i] ≥ x
11      if i < j
12          exchangeA[i]withA[j]
13      else
14          return j
```

Answer the following questions. Show your work to receive partial credit. Assuming $A[p \dots r]$ contains at least two elements, give a careful argument that the procedure HOARE-PARTITION is correct and prove statements b, c, and d.

(a) (5 points) Demonstrate the operation of HOARE-PARTITION on the array $A = \langle 21, 6, 2, 11, 4, 7, 8, 12, 5, 9, 19, 13 \rangle$, showing the values of the array and auxiliary values after each iteration of the while loop in lines 4-14.

(b) (5 points) The indices i and j are such that we never access an element of A outside the subarray $A[p \dots r]$.

(c) (5 points) When HOARE-PARTITION terminates, it returns a value j such that $p \leq j < r$.

(d) (5 points) Every element $A[p \dots j]$ is less than or equal to every element of $A[j+1 \dots r]$ when HOARE-PARTITION terminates.

Answer:

a)

The first while loop (outer while) has swap when j = 12 and i =1 so we have A= [13, 6, 2, 11, 4, 7, 8, 12, 5, 9, 19, 21].
The second outer while loop ends when j = 11 and i = 12 and in this case no swap occurs because i > j return j = 11 In the second outer while loop, we increase from 1 to 12 and A[12]= 21 >= 21 is True, so it's done with this i inner loop and we start comparison between i and j. Since i > j, it returns j = 11.

b)

This is correct, the indices i and j never have access to the element outside its subarray.

As per partitioning we want every element to the left of pivot value to be less than or equal to the pivot and greater than on right side of pivot.

So we will move i marker untill we get element which is greater than or equal to pivot. We do the same with the j marker untill we find an element that is less than or equal to pivot.

When $i < j$ we swap the elements since both the elements are in wrong part of array.

Now if i is not less than j,that means now there is no element in between to swap so we return j position.

So now the array after partitioned lower half is from (start to j) while the upper half is from (j+1 to end).

Therefore the two indeces remain partitioned and separate from each other.
c)
This is true because the value of j will always be between the low(p) and high(r) values of the array. J is greater than or equal to p since it might be the initial value in the array.
d)
This is true since the elements in $A[p \dots j]$ will be the elements which have been partitioned in the lower half. The elements of A[j+1 ... r] are in the right side of the pivot. Thus, the values of A[p...j] will always be less than or equal to the elements of A[j+1 ... r].