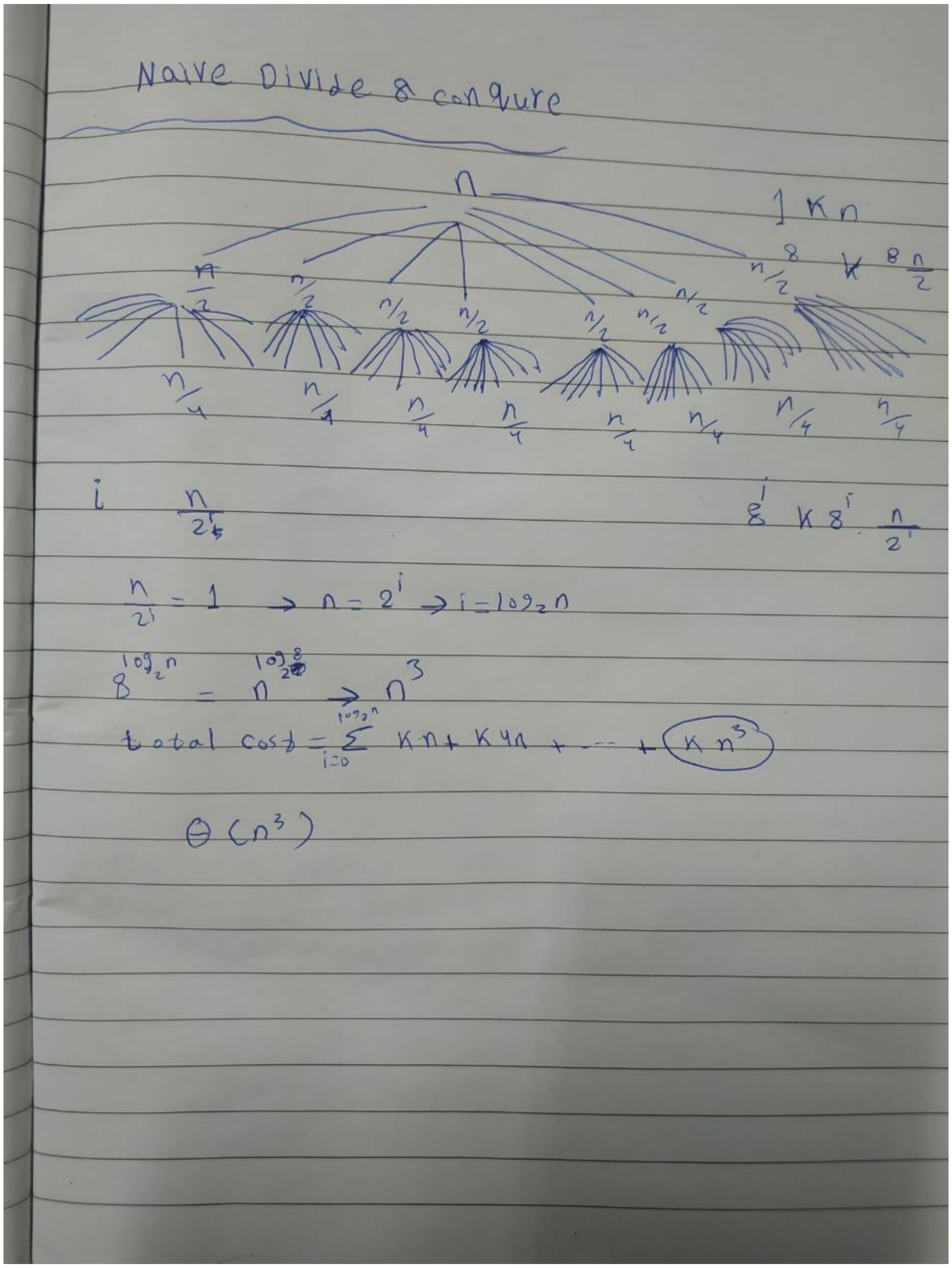
Mohamed AbdelHalem_1180182

Assignment_2

Questions Answers:

1)Complexity of Naïve Approach is O(N^3) As it uses 3 for loops on N

2)o(N^3)



Naive Divide & conqure

$1 \quad Kn$

$8 \quad K\, 8\, \frac{n}{2}$

$i \quad \frac{n}{2^i} \qquad 8^i \quad K\, 8^i \cdot \frac{n}{2^i}$

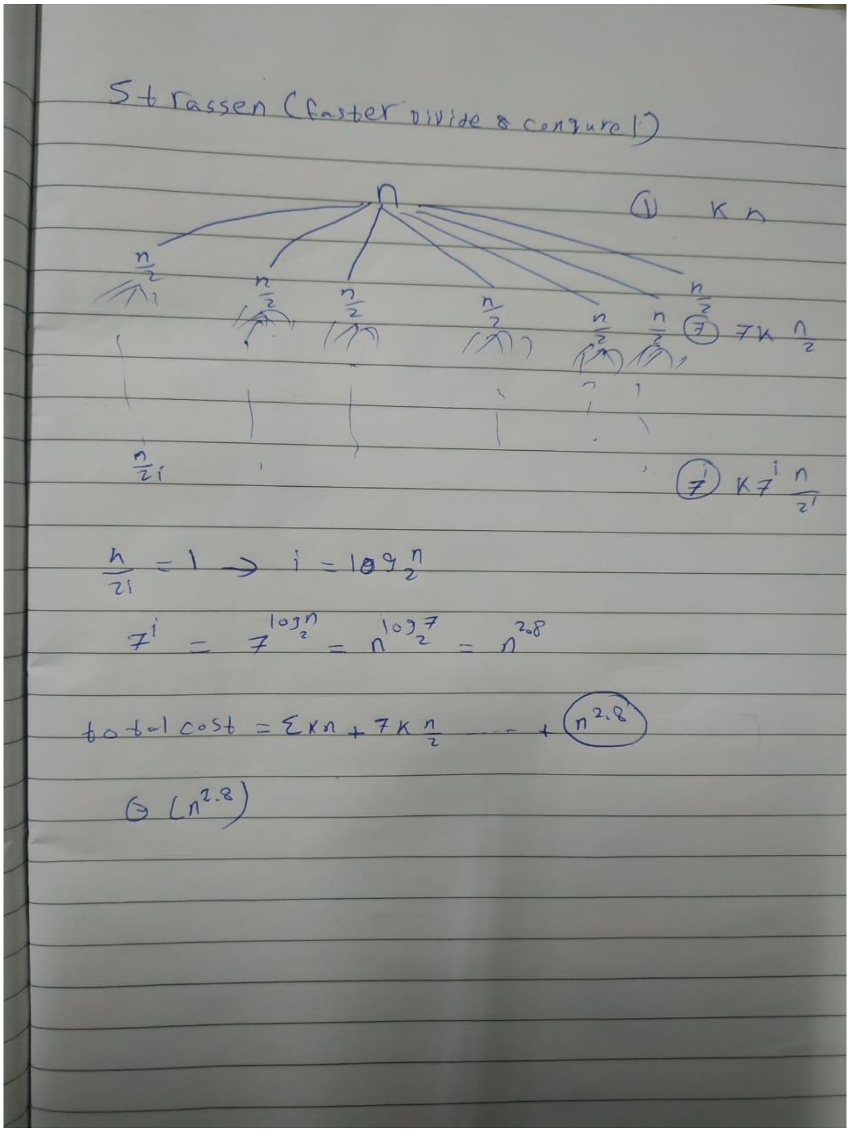
$\frac{n}{2^i} = 1 \rightarrow n = 2^i \rightarrow i = \log_2 n$

$8^{\log_2 n} = n^{\log_2 8} \rightarrow n^3$

total cost $= \sum_{i=0}^{\log_2 n} Kn + K4n + \dots + Kn^3$

$\Theta(n^3)$

3)O(N^2.8)



Divide and Conquer for Matrix Multiplication:

The way I solved this problem:

On understanding Divide and conquer I divided the input matrix to matrices until it's 1x1 matrix to multiply 1 element with the other only and did that using recursion and calling it 8 times to calculate C11,C12,C21&C22 using the formula

I also created a function to split A and B into 4 Matrices and splitted it each time in recursion getting A11,A12,A21,A22 & B11,B12,B21,B22

And after each recursion call is done a matrix is returned for example if we have 8x8 matrix the first call returns 4x4 matrix to add it to the next call which is also 4x4 matrix and the process repeats so at the end we have 4x4 matrix in each Cij

Which leads us to an 8x8 matrix

Also I had to check that the input N is a power of 2 to keep our algorithm working well and I did check that using an equation I found that get the nearest rounded number which is a power of 2

Suppose that we partition each of A, B, and C into four $n/2 \times n/2$ matrices

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}, \quad C = \begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix}, \tag{4.9}$$

so that we rewrite the equation $C = A \cdot B$ as

$$\begin{pmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \cdot \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}. \tag{4.10}$$

Equation (4.10) corresponds to the four equations

$$C_{11} = A_{11} \cdot B_{11} + A_{12} \cdot B_{21}, \tag{4.11}$$

$$C_{12} = A_{11} \cdot B_{12} + A_{12} \cdot B_{22}, \tag{4.12}$$

$$C_{21} = A_{21} \cdot B_{11} + A_{22} \cdot B_{21}, \tag{4.13}$$

$$C_{22} = A_{21} \cdot B_{12} + A_{22} \cdot B_{22}. \tag{4.14}$$

Pseudo Code:

MatrixMult(A,B,N):

First check if N Is power of 2

If it's

Then call Calculate(A,B,N)

If not get the nearest rounded power of 2 from the equation

Nearest power of 2=pow(2,ceil(log(x)/log(2)))

Now get the padding size which is

Pad-size=Neareast power of 2-N

Now pad the matrix with zeros using the pad-size calculated

Now we have Apadded,Bpadded

And call Calculate(Apadded,Bpadded,N)

Calculate(A,B,N)

Check if A length is one and B length is one

If so return A*B

A11,A12,A21,A22=SplitMatrix(A)

B11,B12,B21,B22=SplitMatrix(B)

C11=Calculate(A11,B11,N)+Calculate(A12,B21,N) ….

And do the same as from the equations I left above

Now at we put C11,C12,C21,C22 in a matrix

```
   c = np.vstack((np.hstack((c11, c12)), np.hstack((c21, c22))))
Return c
```

Bonus Problem

Here we use only 7 recursive calls instead of 8

To calculate P1,2,3,4,5,6,7

And then get C11,C12,C21,C22

Pseudo Code:

Check on A length and B if one return A*B

SplitMatrix(A)

SplitMatrix(B)

$$
\begin{aligned}
S_1 &= B_{12} - B_{22} , \\
S_2 &= A_{11} + A_{12} , \\
S_3 &= A_{21} + A_{22} , \\
S_4 &= B_{21} - B_{11} , \\
S_5 &= A_{11} + A_{22} , \\
S_6 &= B_{11} + B_{22} , \\
S_7 &= A_{12} - A_{22} , \\
S_8 &= B_{21} + B_{22} , \\
S_9 &= A_{11} - A_{21} , \\
S_{10} &= B_{11} + B_{12} .
\end{aligned}
$$

Note here on calculating P1 TO 7 we call the function matrixmult_fast so recursion is done 7 times

$$
\begin{aligned}
P_1 &= A_{11} \cdot S_1 &&= A_{11} \cdot B_{12} - A_{11} \cdot B_{22} , \\
P_2 &= S_2 \cdot B_{22} &&= A_{11} \cdot B_{22} + A_{12} \cdot B_{22} , \\
P_3 &= S_3 \cdot B_{11} &&= A_{21} \cdot B_{11} + A_{22} \cdot B_{11} , \\
P_4 &= A_{22} \cdot S_4 &&= A_{22} \cdot B_{21} - A_{22} \cdot B_{11} , \\
P_5 &= S_5 \cdot S_6 &&= A_{11} \cdot B_{11} + A_{11} \cdot B_{22} + A_{22} \cdot B_{11} + A_{22} \cdot B_{22} , \\
P_6 &= S_7 \cdot S_8 &&= A_{12} \cdot B_{21} + A_{12} \cdot B_{22} - A_{22} \cdot B_{21} - A_{22} \cdot B_{22} , \\
P_7 &= S_9 \cdot S_{10} &&= A_{11} \cdot B_{11} + A_{11} \cdot B_{12} - A_{21} \cdot B_{11} - A_{21} \cdot B_{12} .
\end{aligned}
$$

C11 = p5 + p4 - p2 + p6

C12 = p1 + p2

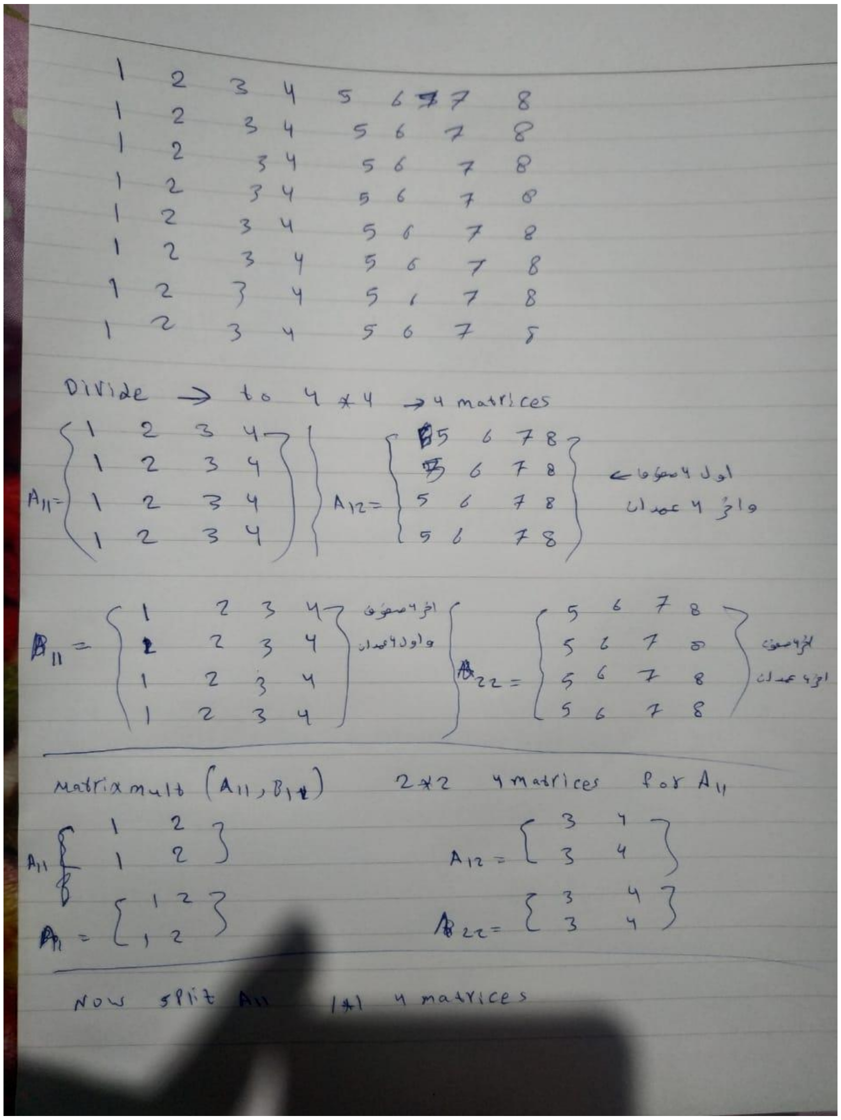
C21 = p3 + p4

C22 = p1 + p5 - p3 – p7

Return C

***Note I might have repeated a function just the keep the input/output down as it's but it could be 1**

General Explanation for the recursion (first problem as example):



$A_{11} = \{1\}$ $A_{12} = \{2\}$ $A_{21} = \{3\}$ $A_{22} = \{4\}$

len = 1 Now as if B is same As A

$A_{11} * B_{11} = 1$

then go to previous call
do matrix mult (A_{12}, B_{21}) → here we split A_{12}
the one before we did split A_{11} so After split
. A_{12} & B_{21} from Prev we get $A_{12} * B_{21}$
then go to previous call & continue until we reach
C & put items in it then continue

Sources used in this problem:

https://shivathudi.github.io/jekyll/update/2017/06/15/matr-mult.html