Mohamed Khelif

CSC 220

Nooreddin Naghibolhosseini

Assignment 2 Part B

**(4.2-3)**

Strassens algorithm applies the divide and conquer technique in order to multiply n x n matrices. Therefore the matrices must be in powers of 2 for example a 3x3 matrix must be extended to a 4x4 matrix. This can be achieved by adding 0’s to the end of the matrix

Since n cannot be extended twice or more, it increases constantly there fore the run time is Θ(n^lg7)

**(4-2-5)**

T(n) = 132454T(n/68)+n^2

T(n) = Θ(n^2.795128)

T(n) = 143640t(n/70)+n^2

T(n) = Θ(n^2.795128)

T(n) = 155424T(n/72)+n^2

T(n) = Θ(n^2.795147)

Strassens algorithm runs in 2.81 all these algorithms perform better then Strassens

**(4.3-8)**

T(n)<=4(cn/3)log^4-n)+n

<=cnlog^4n+n

<=cnlog^4-3n

<=Cnlog^4-n

**(4.4-7)**

Assuming that T(k) < dk2 − ek

T(n) = 4T(bn/2c) + cn

< 4(d(n/2)2 − e(n/2)) + cn

= dn2 − 2en + cn

= dn2 − en − (en − cn)

= dn2 − en − n(e − c)

< dn2 − en if e > c

**(6.4-4)**

N! > (n/e)^n

lg(n!) > lg(n/e)^n

=> h>= nlg(n/e)

=> h >= n(lgn/lge)

=> h>= nlgn

= Ω(nlgn)

**(6.5-8)**

HEAP-DELETE(A, i):

A[i] = A[A.heap-A.size]

heap - A.size <- heap - A.size-1

Key = A[i]2

if(key <= A[Parent(i)]{

max-heapify(A,i)

}else{

while(i>1 and A[Parent(i)] < key){

A[i] = A[Parent(i)]

i = Parent(i)

}