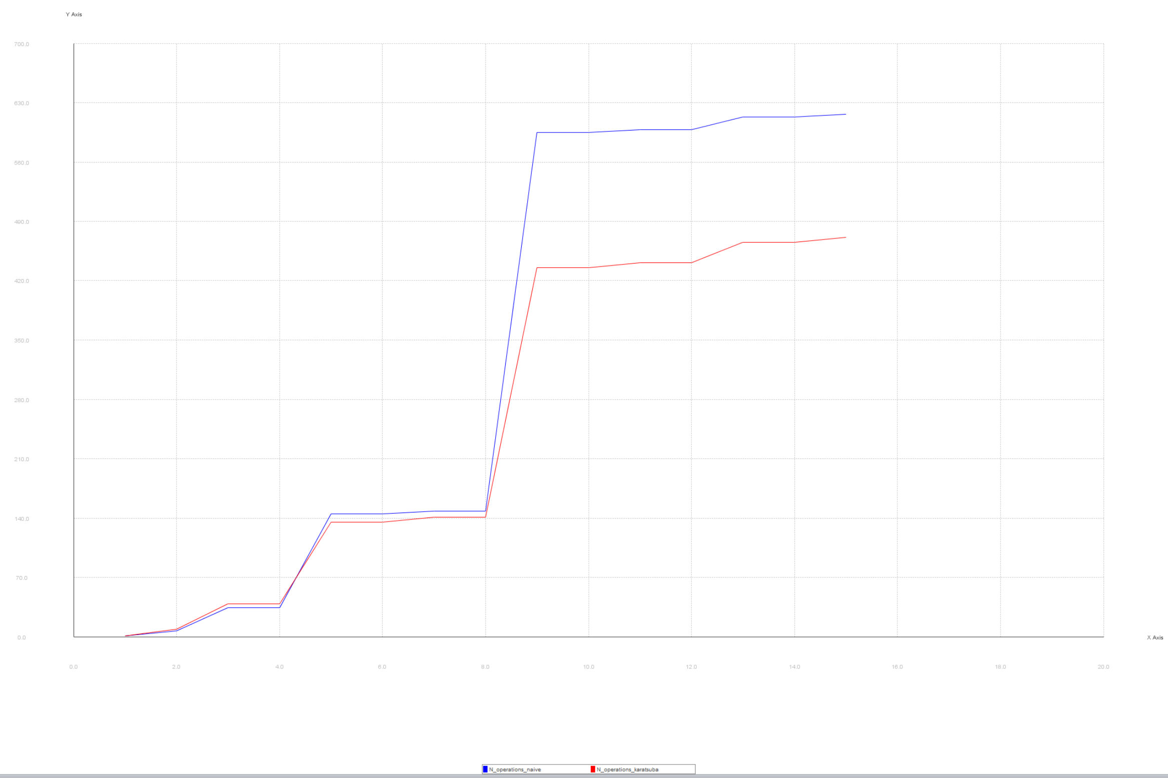

Generally: As more recursion occurs (size gets larger than 8), the larger difference shows up in between naïve and Karatsuba algorithm. At size = 8 , the naïve algo jump from operation cost 148 to 595 and the Karatsuba algorithm increase the cost from 141 to 435. When at low size (size smaller than 8) both algorithm seems to have a similar cost. (Note that in case n = 1 the operation is 1 for both) As mentioned, increasing the size will lead to more cost of running time in naïve algorithm compare to Karatsuba algorithm.

(1) The **naïve algorithm** is by using equation:

X =X1(*2^(N/2)) + X2

Y = Y1(*2^(N/2)) + Y2

XY = (X1*Y1)*(*2^(N) +(X1*Y2 + X2* Y1)*((*2^(N/2))) +X2*Y2

Running time:

T(n) = (4* T(n/2))+ 3 ;(which 3 is O(n), a linear time) By master theorem:

take a = 4, b =2, k = $\log_b a$ = 2, f(n) = n.

Here, n^($\log_b a$) = n^2 ;

where n^($\log_b a$) > f(n) fit in the **case one: (for more detail of theorem check question 2)**

T(n) = O($n^{(\log_b a)}$) = O(n^2);

(2) The **Karatsuba algorithm** is equation:

$XY = (X1*Y1)*(*2^(N)) + ((X1*Y1) + (X2*Y2) – (X1 – X2)(Y1 -Y2))*(*2^(N/2) + X2*Y2$

Running time:

$T(n) = (3* T(n/2))+ 6$; (which 6 is O(n), a linear time) By master theorem:

take a = 3, b =2, k = $\log_b a$ = 1.58, f(n) = n so d =1. where n^ ($\log_b a$) > f(n), fit in the **case one:**

$\mathbf{T(n) = O(n^{(\log_b(a))}) = O(n^{1.58});}$

However, the Karatsuba algorithm require 6 operation compare to naïve algorithm only has 3. Therefore at the low size , Karatsuba algorithm cost more than naive algorithm. As the size (n) increase, Karatsuba require less recursive than naïve algorithm, with allow Karatsuba algorithm to grow less slowly than naïve algorithm

PART 2 (Question 1)

1) Approach: obtain “a” and “b”
2) Calculate n ^ $\log_b a$ (which k = $\log_b a$)
3) Compare with f(n)
4) Apply to master theorem

f(n) < $\log_b a$ =case 1, cause need to subtract epsilon

f(n) > $\log_b a$, could be case 3 , check if a*f(n/b) <= c*f(n)

Solution:

a) T(n) = 25*T(n/5) + n:

   a = 25, b = 5, so $\log_b a$ = 2

   f(n) =n , f(n) < n^2, fit in **case 1** which f(n) = O(n^(2 - epsilon)) (epsilon = $\log_b a$ -1 =1, so epilson >0), so $\mathbf{T(n) = \theta(n^2)}$

b) T(n) = 2*T(n/3) +n*log(n):
   a=2, b = 3, so $\log_b a$ =0.63, n^0.63 smaller than n.
   f(n)= n*log(n),
   f(n) > n^0.63, could be in **case 3, next check the constant c**
   a(n/b) log(n/b) <= c*f(n)
   = 2(n/3) log(n/3) <= c*f(n)
   = (2/3) nlog(n/3) <= c* f(n)

This holds for c = 2/3 which c <1 , holds!

f(n) = nlog(n)= omega($n^{k + episilon}$) (with 1- $log_3 2$>epilson >0)

Apply **case 3:**

T(n) = θ(f(n)) = θ(nlog(n))

c) T(n) = T(3n/4) + 1 :

a = 1, b = 4/3 so $log_b a$ =0

f(n) =1, f(n) == n^ ($log_b a$), this is **case 2 , evenly distributed** which f(n) = **O(1), T(n) = θ((n^(** $log_b a$**))* log(n) = θ(log(n))**

d) T(n) = 7*T(n/3) +n^3:

a = 7, b = 3 so $log_b a$ =1.77

f(n) = $n^{3.}$, f(n) > n ^1.77,

check for the constant c :

$7(n/3)^3$<= c* $n^{3.}$

Holds with c = 7/9. which is less than 1

f(n) = n^3= omega($n^{k + episilon}$) (with epilson >0)

so apply **case 3:**

T(n) = θ(f(n)) = θ(n^3)

e) T(n) = T(n/2) +n(2 – cos(n)):

a=1, b =2 , $log_b a$ = 0

f(n) = n(2 – cos(n))

relationship between f(n) and n^0 **could not be defined** in any case of master therom.

PART 3(Question 1)

$T_A(n) = 7T_A(n/2) + n^2$

$T_B(n) = \alpha T_B(n/4) + n^2$

⇨ Calculate case of T_A,

a = 7, b = 2 so $log_b a$ =2.81

f(n) = n^2, f(n) < n ^2.81, there for T_A is **case1 master theorem** with :

f(n) = O(n^(2.81 - epsilon)) (with epilson >0)), so T(n) = θ($n^{2.81}$)

1) Assume $T_B(n)$ is also **case1 master theorem:**

a = α, b = 4 so $log_4 \alpha$ = k

$f(n) = n^2$, f(n) has to be smaller than n ^k according to assumption as case1 ,

- ⇨ $n^k > n^2$
- ⇨ $\log_4\alpha > 2$
- ⇨ $\alpha > 2^4$
- ⇨ $\alpha > 16$;

since $T_B(n)$ has to be asymptotically faster than $T_A(n)$, so

$T_B(n) < T_A(n)$

- ⇨ $\theta(n^\wedge(\log_4\alpha)) < \theta(n^\wedge(\log_2 7))$
- ⇨ $\log_4\alpha < \log_2 7$
- ⇨ $\log_4\alpha < 2 * \log_4 7$
- ⇨ $\alpha < 49$

For Case 1 to be true : $16 < \alpha < 49$.

2) Assume $T_B(n)$ is also **case2 master theorem:**
   $a = \alpha$, $b = 4$, so $\log_4\alpha = k$
   $f(n) = n^2$, f(n) has to be equal to n ^k according to assumption as case2:

- ⇨ $n^k = n^2$
- ⇨ $K = 2$
- ⇨ $\log_4\alpha = 2$
- ⇨ $\alpha = 16$;

since $T_B(n)$ has to be asymptotically faster than $T_A(n)$, so

$T_B(n) < T_A(n)$

- ⇨ $\theta(n^\wedge(\log_4 16)) < \theta(n^\wedge(\log_2 7))$

this condition doesn't hold, since it's not asymptotically faster

3) Assume $T_B(n)$ is also **case3 master theorem:**
   $a = \alpha$, $b = 4$, so $\log_4\alpha = k$
   $f(n) = n^2$, f(n) has to be larger than n ^k according to assumption as case3:

- ⇨ $n^k < n^2$
- ⇨ $K < 2$
- ⇨ $\log_4\alpha < 2$
- ⇨ $\alpha < 16$;

   In addition the constant c should be smaller than 1:

- ⇨ a*f(n/b) <= c*f(n)
- ⇨ $\alpha * (n/2)^2 <= c * n^2$
- ⇨ $\alpha/4 < 1$ (since c is less than 1)
- ⇨ $\alpha < 4$

since $T_B(n)$ has to be asymptotically faster than $T_A(n)$, so

$T_B(n) < T_A(n)$

- ⇨ $\theta(f(n)) < \theta(n^{\wedge}(\log_2 7))$
- ⇨ $n^2 < n^{\wedge}(\log_2 7)$

this condition doesn't hold.

In conclusion, for asymptotically faster to holds, algorithm B has to be a master theorem case 1, and the alpha range:

16 < α < 49.

The largest int here is 48.