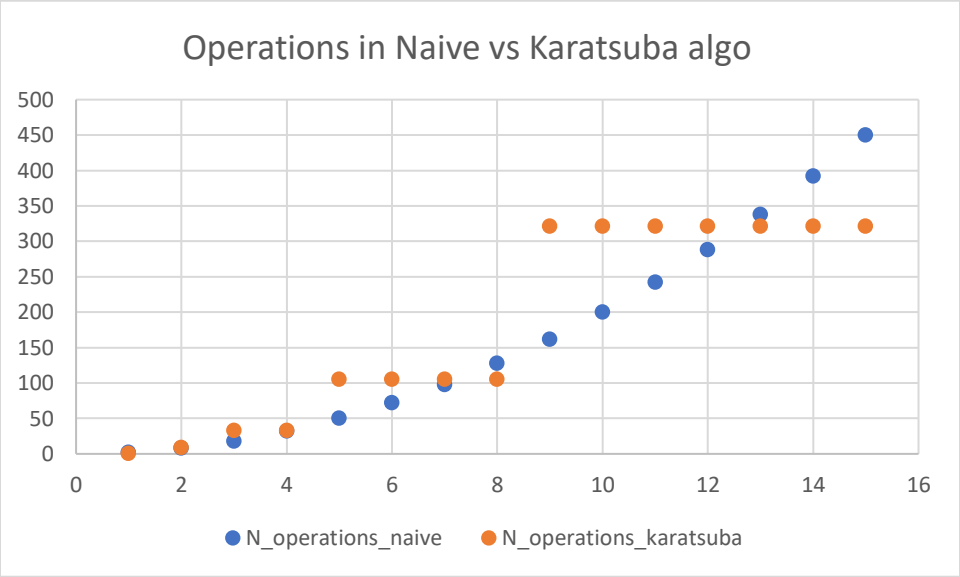
David Ivanov ID:260806837



The relationship between the number of operations and the magnitude of the number operated on are very distinct when comparing the naïve algorithm and the Karatsuba.

The naïve algorithm increases directly depending on the size of the number operated on. If the number is bigger therefore there will be more operations to perform. The growth of the function as the number increases appears to be quadratic.

The Karatsuba algorithm increases depending on the size of the integers on which the operation of multiplying is performed upon. The step function is the best way to represent it and at values like 3, 5, 9 where the size of the binary number increases the step function jumps.

On small integers the naïve algorithm appears to be more efficient, but as the sizes increase the Karatsuba will be more efficient. On a larger size more numbers can be represented making the steps wider and on these wide steps the naïve algorithm will grow very fast.

Part 2

David Ivanov
ID: 260806837

2. a) $T(n) = 25 \cdot T(\frac{n}{5}) + n$

$a = 25$ $\quad K = \log_5 25 = 2$

$b = 5$

$f(n) = n$

we know $f(n)$ has $O(n)$

? $f(n) \in O(n^{2-\epsilon})$ let $\epsilon = 1$ and $\epsilon > 0$ then $f(n) = O(n)$ ✓

$T(n) = \Theta(n^2)$

b) $T(n) = 2T(\frac{n}{3}) + n\log n$

$a = 2$

$b = 3$ $\quad K = \log_3(2)$

$f(n) = n\log(n)$

$f(n)$ has $\Theta(n\log(n))$ $\quad f(n) \notin \Theta(n^{\log_3(2)} \log^p(n))$

no such value of p exists

$\Rightarrow$ we cannot use master theorem

c) $T(n) = T(\frac{3n}{4}) + 1$

$a = 1$ $\quad K = \log_{\frac{4}{3}} 1 = 0$

$b = \frac{4}{3}$

$f(n) = 1 \in O(1)$

? $f(n) \in O(n^{0-\epsilon})$ since $\epsilon > 0$ it's impossible to find a value of ϵ

$\Rightarrow$ we cannot use master theorem

d) $T(n) = 7 \cdot T(\frac{n}{3}) + n^3$

$a = 7$ $\quad k = \log_3 7 \cong 1.77\ldots$

$b = 3$

$f(n) = n^3 \vee O(n^3)$; $\Omega(n^3)$, $\Theta(n^3)$ (is element of)

? $f(n) = O(n^{\log_3 7 - \epsilon})$ $\quad \epsilon = \log_3 7 - 3 < 0$ ✗

$\hookrightarrow = -1,229\ldots$

Case 1 doesn't work check case 3

? $f(n) = \Omega(n^{\log_3 7 + \epsilon})$ $\quad \epsilon = 3 - \log_3 7 > 0$ ✓

$7\left(\frac{n}{3}\right)^3 \leq c\, n^3$ $\quad$ if $c = \frac{7}{27}$ at $n_0 = 0$

this is true

$\Rightarrow \underline{T(n) = \Theta(n^3)}$

e) $T(n) = T(n/2) + n(2 - \cos n)$

$a = 1$ $\quad k = \log_2 1 = 0$

$b = 2$

$f(n) = n(2 - \cos n) = 2n - n\cos n \vee O(n)$ (is element of) $\Omega(n)$ and $\Theta(n)$

Case 3

? $f(n) \in \Omega(n^{0+\epsilon})$ yes have $\epsilon = 1 > 0$ so it works

$a f(\frac{n}{b}) \leq C(n(2 - \cos(n))$

$n - \frac{n\cos(n/2)}{2} \leq C(n(2 - \cos(n))$

for $n_0 = 0$ and $C = 5$ the above is true

$T(n) = \Theta(2n - n\cos n) = \underline{\Theta(n)}$

3. $T_A = 7T_A\left(\frac{n}{2}\right) + n^2$

$a = 7$ $\quad k = \log_2 7 \cong 2.80\ldots$

$b = 2$

$f(n) = n^2 \in O(n^2)$

? is $f(n) \in O(n^{\log_2 7 - \epsilon})$

$\epsilon = \log_2 7 - 2 > 0$ so $T_A(n) = \Theta(n^{\log_2 7})$

$T_B = \alpha T_A\left(\frac{n}{4}\right) + n^2$

$a = \alpha$ $\quad k = \log_4 \alpha$

$b = 4$

$f(n) = n^2 \in O(n^2)$

for $f(n) \in O(n^{\log_4 \alpha - \epsilon})$

then $\epsilon = \log_4 \alpha - 2$ so $\alpha > 16$

$T_B(n) = \Theta(n^{\log_4 \alpha})$

so for $T_B(n)$ to be strictly faster than $T_A(n)$

$\log_4 \alpha \leq \log_2 7$

$\Rightarrow \frac{\log(\alpha)}{\log(4)} \leq \frac{\log(7)}{\log(2)} \Rightarrow \log(\alpha) \leq \frac{\log(7)\log(4)}{\log(2)}$

$\Rightarrow \alpha \leq 10^{\left(\frac{\log(7)\cdot\log(4)}{\log(2)}\right)}$

$\Rightarrow \alpha \leq 49$

α must be less than 49