Design an algorithm to compute the binary representation of 10n in O(n^log23) time.

Import Integer Multiplication function from Lecture Sept 15

(number)2 is means in binary form

Input: n is an integer. n >= 1

Output: (10n)2

Procedure Pt(n):

If n =1 return (1010)2 //in binary

x = Pt(⌊10n/2⌋) // in binary

y = Integer Multiplication (x, x) // in binary

If n is even:

Return y

Else:

Return y\*(1000)2 + y\*(10)2

Proof of Correctness:

BC: n =1

Return 1010 which is 10 in binary

IH: Pt(n) return 10n in binary for 1≤ n ≤ k

IS: n =k+1

By IH, x = (10⌊n/2⌋)2

By correctness of IH,

Y = (102⌊n/2⌋)2

If n is even,

102⌊n/2⌋ = 10n

Return (10n)2

Else n is odd,

102⌊n/2⌋ = 10n-1

Return ((10n-1)2 \* (10)2)2

Return (10n)2

Runtime Analysis:

In Pt(n), the dominating factor is the Integer Multiplication function which is O(n^log23) If n is integer and n ≥ 1

Since we have a recursion Tree right here, the depth is log2n

So the total runt time (all the log is log base 2)

=

And ≤

= Cn^log3 () = which is upper bonded by O(n^log23)