

# Log Factorial

## Theoritical proof

In order to determine the growth rate of log factorial function, let’s compute it :

So the computationnal cost is 0(nlog(n)).

More points would be required to determine if the time complexity is 0(nlog(n)).

# Sum of Log Factorial

## Theoritical proof

In order to determine the growth rate of the sum of the log factorial function, let’s compute it :

Hence the computational cost is 0(nlog(n !)) which is equal by using the previous expression to .

# Fibonacci

## Theoritical proof

The sequence is models as F(n) = F(n-1) + F(n-2) thus the time function to calculate F(n) is the sum of the time to compute F(n-1) plus the one for F(n-2) plus the time to add them 0(1).

This scheme can be represented by a recursion tree with a depth of n, at each node there is two leafs hence we can induct that the time complexity is .

The log-log scale allows us to visualize a time complexity that could look like to but as it not a straight line we could rather think it is an exponential cost : .

