I collaborated with: W3023339

Problem
Suppose you are choosing between the following three algorithms:

1. Algorithm A solves problems by dividing them into five subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.
2. Algorithm B solves problems of size n by recursively solving two subproblems of size $n-1$ and then combining the solutions in constant time.
3. Algorithm C solves problems of size n by dividing them into nine subproblems of size $n/3$, recursively solving each subproblem, and then combining the solutions in $O(n^2)$ time.

What are the running times of each of these algorithms (in asymptotic notation) and which would you choose?
Solution

1. $T(n) = 5T(\frac{n}{2}) + cn = O(n^{\log_2 5})$
2. $T(n) = 2T(n-1) + c = O(2^n)$ as every step makes two recursive calls of size $n-1$. If we think about this as a tree, $n-1$ eventually approaches zero and 2^n leaves are created.
3. $T(n) = 9T(\frac{n}{3}) + cn^2 = O(n^2 \log n)$

I would choose Algorithm C as $2^n > n^{\log_2 5}$ because exponentials grow quicker than polynomials and $n^{\log_2 5} > n^2 \log n$ as:

$$\lim_{n\to\infty} \frac{n^2 \log n}{n^{\log_2 5}}$$
$$\lim_{n\to\infty} \frac{\log n}{n^{0.32}} = 0$$

Polynomials grow quicker than logarithmic functions.