Divide and Conquer Recursions

Rounding errors
Suppose that the time for \texttt{Mergesort} is
\[
M(n) = M\left(\left\lceil \frac{n}{2} \right\rceil \right) + M\left(\left\lfloor \frac{n}{2} \right\rfloor \right) + 100n \ \mu s \\
M(1) = 100 \ \mu s
\]
but we approximate it by
\[
M'(n) = 2M'\left(\left\lfloor \frac{n}{2} \right\rfloor \right) + 100n \ \mu s \\
M'(1) = 100 \ \mu s
\]
These agree when \( n = 2^m \) is a power of 2. Both are increasing functions, and \( n \lg n \) is “smooth,” so the graphs stay within a constant factor of each other.

Hybrid algorithms
Suppose \texttt{Mergesort} has the times given above, and \texttt{Exchange sort} runs in time
\[
E(n) = 20n^2 \ \mu s
\]
We may combine the two algorithms to produce an even faster one: “Hybrid Sort.” We will pick a threshold \( t \). It will be in the interval where \texttt{Exchange sort} is faster than \texttt{Merge sort}.
When an instance has size \( n \geq t \), we divide it in two as for Mergesort, sort the two halves using Hybrid Sort, and merge them together using the normal \texttt{Merge} procedure of \texttt{Mergesort}.
When it has size \( n < t \), we sort it using \texttt{Exchange sort}.
This gives a recursion
\[
H(n) = \begin{cases} 
H\left(\left\lceil \frac{n}{2} \right\rceil \right) + H\left(\left\lfloor \frac{n}{2} \right\rfloor \right) + 100n \ \mu s & \text{if } n \geq t \\
E(n) = 20n^2 \ \mu s & \text{if } n < t.
\end{cases}
\]
What is the optimal value of \( t \)? Assume \( n \) is even. At the boundary, we have \( n \geq t \) but \( n/2 < t \), so \( H(n) \) is expressed using the top line in the definition, but \( H(n/2) \) is expressed using the bottom line. Then \( H(n/2) = 20(n/2)^2 = 5n^2 \) so \( H(n) = 2H(n/2) + 100n = 10n^2 + 100n \). We want Hybrid Sort to be faster than Exchange Sort, so
\[
H(n) \leq E(n) \iff 10n^2 + 100n \leq 20n^2 \iff 100n \leq 10n^2 \iff 10 \leq n.
\]
Thus \( t = 10 \).