Algorithms and Data Structures: Homework #4

Due on March 2, 2020 at 23:00 PM

Henri Sota

Problem 4.1

d) Based on the results from (b) and (c), how would you choose k in practice? Briefly explain.
In order for Merge Sort optimized with Insertion Sort algorithm to perform better than standard Merge Sort, k must be chosen in a way that runtime of the hybrid algorithm to be always lower than the runtime of standard Merge Sort. k must vary depending on the size of the element array and must be a value such that for that particular input size, Insertion Sort running on $\frac{n}{k}$ sublists should be faster than Merge Sort running on those sublists. Bringing in the equation from c). Assuming $k = \Theta(\log n)$:

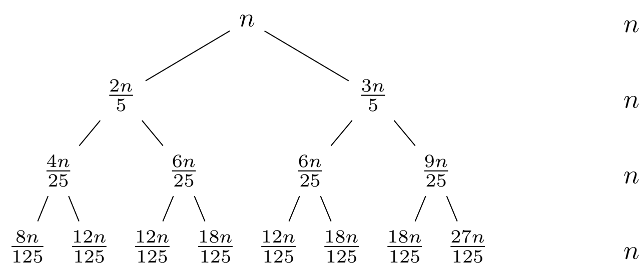
$$\begin{aligned}
\Theta(nk + n\log(\frac{n}{k})) &= \Theta(nk + n\log n - n\log k) \\
&= \Theta(n\log n + n\log n - n\log(\log n)) \\
&= \Theta(2n\log n - n\log(\log n)) \\
&= \Theta(n\log n)
\end{aligned}$$

Therefore k should be chosen to be less than or equal to $\Theta(\log n)$ or $\log_2(n)$.
Example: $n = 100000 \rightarrow k = \log_2 100000 \approx 16$

Problem 4.2

Use the substitution method, the recursion tree, or the master theorem method to derive upper and lower bounds for $T(n)$ in each of the following recurrences. Make the bounds as tight as possible. Assume that $T(n)$ is constant for $n \leq 2$.

e) $T(n) = T(2n/5) + T(3n/5) + \Theta(n)$
Recursion tree method can be used:



$f(n) = \Theta(n) \implies f(n) = n$
In order to calculate a tight bound, the height of the tree has to be calculated based on the longest path and the shortest path from the root to a leaf. In this case, the longest path is calculated by following the rightmost node on each level and the shortest path is calculated by the following the leftmost node on each level.

$$h_l = \log_{\frac{5}{3}} n = \frac{\log n}{\log \frac{5}{3}} = c_l \log n = O(\log n)$$
$$h_s = \log_{\frac{5}{2}} n = \frac{\log n}{\log \frac{5}{2}} = c_s \log n = \Omega(\log n)$$

Each level of the recursion tree has a cost n. The total cost is sum of the cost of each level.

$$\sum_{i=0}^{h} n = n * \sum_{i=0}^{h} 1 = n * h = n * \log n$$

$T(n) = \Theta(n \log n)$