John Boren
12.6.2018
CSCE A311 HW#4

Problem #1: Insertion sort on small arrays in merge sort.

1) Show that the n/k sublists, each of length k, can be sorted by insertion sort in $\Theta(nk)$ worst-case time.

Note that sorting each list takes ak^2+bk+c for some constants a,b,c. We have n/k of those, therefore:

$$\frac{n}{k}(ak^2+bk+c) = ank+bn+\frac{cn}{k} \sim \Theta(nk)+\Theta(n)+\Theta(\tfrac{n}{k}) \sim \Theta(nk)$$

2) Show that the sublists can be merged in $\Theta(n\lg(\frac{n}{k}))$ worst-case time.

- merging n/k sublists into $n/2k$ sublists takes $\Theta(n)$ worst-case time.
- merging $n/2k$ sublists into $n/4k$ sublists takes $\Theta(n)$ worst-case time.
- ... and so on
- merging 2 sublists into list takes $\Theta(n)$ worst-case time.
- We have $\lg(\frac{n}{k})$ such merges, so merging n/k sublists into one list takes $\Theta(n\lg(\frac{n}{k}))$

3) The largest value of k.

- In order for $\Theta(nk+n\lg(\frac{n}{k}))=\Theta(n\lg n)$, either $nk=\Theta(n\lg n)$ or $n\lg(\frac{n}{k})=\Theta(n\lg n)$. From these we get the largest asymptotic value for k is $\Theta(\lg n)$. If we substitute, we get: $\Theta(n\lg n+n\lg(\frac{n}{\lg n}))=\Theta(n\lg n)$

If $k=f(n)>\lg n$, the complexity will be $\Theta(nf(n))$, which is a larger running time than merge sort.

4) In practice, k should be the largest list length on which insertion sort is faster than merge sort.