11. Sorting Algorithms :: Merge Sort :: Time Complexity Analysis (continued)

There are various methods for solving recurrence relations, but one approach that will often yield a solution is to examine enough examples until a pattern emerges:

11. Sorting Algorithms :: Merge Sort :: Time Complexity Analysis (continued)

At this point a fairly clear pattern emerges:

$$a(n) = 2^k a(n/2^k) + 5nk$$

Eventually, the sizes of the two sublists will be 1 and there will be no list accesses, i.e., $a(1) = 0$. And when will this happen? When $n/2^k = 1$. Solving for k we determine that $k = lg\ n$. Therefore,

$$a(n) = 2^{lg\ n} a(n/2^{lg\ n}) + 5n(lg\ n) = n \cdot a(1) + 5n(lg\ n) = 5n\ lg\ n$$

which can be easily shown to be $O(n\ lg\ n)$.