Adam Frenkel

The Iterative Algorithm

The Recurrence:

$z = 1 \rightarrow T(z) = O(1)$

$z > 1 \rightarrow T(z) = T(z-1) + O(zn)$

Justification:

The recurrence is accurate because when z = 1, no order n work is done.
Finally, when z > 1, one must first merge 2 arrays with a total size of 2n, then 2 arrays with a total size of 3n, then 4n … zn. This is represented by the recurrence, as first one does zn then … 4n, then 3n, then 2n of work.

Solving The Recurrence:

$$
\begin{aligned}
z > 1 \rightarrow T(z) &= T(z-1) + O(zn) \\
&= T(z-2) + O\big((z-1)n\big) + O(zn) \\
&= T(2n + \cdots \; O\big((z-1)n\big) + O(zn) \\
&= O(\sum_{i=2}^{z} i(n)) \\
&= O(n \sum_{i=2}^{z} i) \\
&= O(n(\left(\frac{z(z+1)}{2}\right) - 1)) \\
&= O(nz^2) \\
&= O(z^2)
\end{aligned}
$$

Closed Form:

$z = 1 \rightarrow T(z) = O(1)$

$z > 1 \rightarrow T(z) = O(z^2)$

Divide & Conquer Algorithm

The Solution:
My algorithm is simply the combining part of the standard merge sort algorithm.

The Recurrence:
$z = 1 \rightarrow T(z) = O(1)$
$z > 1 \rightarrow T(z) = zn + 2T(z/2)$

Justification:
When z = 1, no order n work is necessary.
When z > 1, you have to combine every pair of arrays (zn). After doing so, you must combine those arrays which will be half the amount of those original arrays T(z/2). Because these arrays are twice the size though it will come out to 2T(z/2). Giving a total work of $zn + 2T(z/2)$.

Solving the Recurrence:
$z > 1 \rightarrow T(z) = zn + 2T(z/2)$

Plug	zn + 2((zn/2) +2T(z/4))
Chug	zn + zn + 4T(z/4)
Plug	zn + zn + 4((zn/4) +2T(z/8))
Chug	zn + zn + zn + 8T(z/8)
Plug	zn + zn + zn + 8((zn/8) +2T(z/16))
Chug	zn + zn + zn + zn + 16T(z/16)
Pattern	T(z) = 2T(z/2) + 4T(z/4) … + xT(z/x), where (z/x) = 1
Equivalent to	T(z) = zn + zn … + zn, log(z) times
Equivalent to	T(z) = log(z) × zn

Closed Form:
$z = 1 \rightarrow T(z) = O(1)$
$z > 1 \rightarrow T(z) = \log(z) \times zn$