n

1

$O(m)$ m

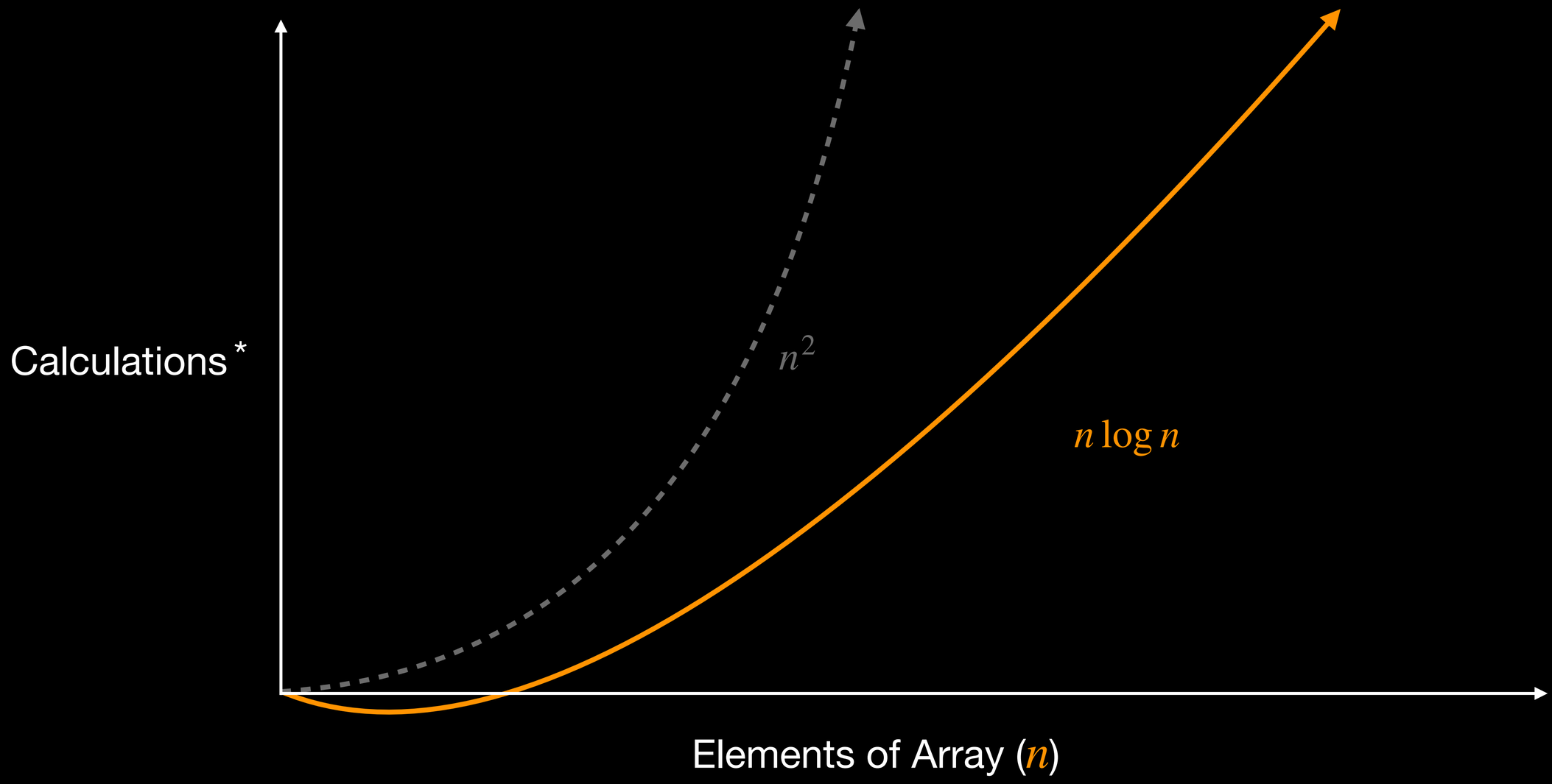
$\log_2(n)$ n

$\log_2(n)$ $2n$

$O(n \log n)$

How efficient is Merge Sort?

- Let our array contain n elements. Divide the full array into equal subarrays, each with 1 element.
- Each merge operation takes $O(m)$, where each subarray has m elements.
- Each time we merge two subarrays, the resulting merged array is double the size of either subarray.
- We can double the size of any subarray at most $\log_2(n)$ before the subarray is of size n.
- So, we perform $\log_2(n)$ merges, which could each take up to $2n$ operations.
- The overall running time is $O(n \log n)$.



* Loose definition