1. The divide-and-conquer paradigm can be described in general terms as consisting of the following three steps: (1) divide, (2) recur, and (3) conquer.

Write down the recurrence relation showing the running time of a generic recursive, divide-and-conquer algorithm:

\[ T(n) = D(n) + kT\left(\frac{n}{r}\right) + C(n) \]

2. Divide-and-conquer. Suppose a recursive divide-and-conquer algorithm partitions the original problem into 5 subproblems, each of size \(n/3\), and that it spends \(\Theta(n)\) time to partition the original problem into subproblems and \(\Theta(n \log n)\) time to combine the subproblem solutions into the solution to the original problem, where \(n\) is the input size.

Write down the recurrence relation showing the running time of this algorithm:

\[ T(n) = 5T\left(\frac{n}{3}\right) + \Theta(n \log n) \]

3. Merge sort. Merge sort is an algorithm that applies the divide-and-conquer paradigm. Merge sort partitions the original problem of size \(n\) into 2 subproblems each of size \(n/2\), and spends \(O(n)\) time total to partition the problem into subproblems and to combine the subproblem solutions into a solution to the original problem. The overall running time of merge sort is \(O(n \log n)\) time.

Write down the recurrence relation showing the running time of merge sort (state any assumptions you make):

\[ T(n) = 2T\left(\frac{n}{2}\right) + O(n) \]

4. QuickSort. Quicksort is an algorithm that applies the divide-and-conquer paradigm. Quicksort partitions the original problem of size \(n\) into 2 subproblems and spends \(O(n)\) time total to partition the problem into subproblems and to combine the subproblem solutions into a solution to the original problem. The overall running time of QuickSort is \(O(n \log n)\) time (average case).

Write down the recurrence relation showing the running time of QuickSort (for the average/expected case):

\[ T(n) = 2T\left(\frac{n}{2}\right) + O(n) \]