



DOPE (/blog/)

Divide and Conquer

🕒 Nov. 11, 2018 🔖 RECURSION (/blog/tag/recursion/?tag=recursion) ALGORITHM (/blog/tag/algorithm/?tag=algorithm) DIVIDE AND CONQUER (/blog/tag/divide-and-conquer/?tag=divide-and-conquer) PYTHON (/blog/tag/python/?tag=python) 👁 680



Become an Author

(/blog/submit-article/)

Download Our App.

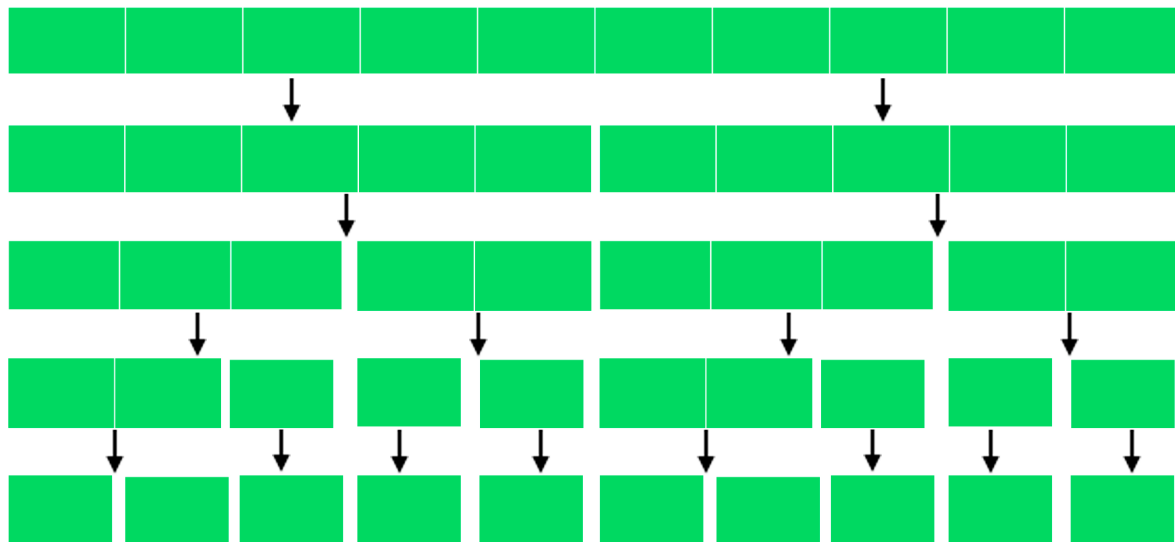


(<https://play.google.com/store/apps/details?id=com.blogsdope&pcampaignid=MKT-Other-global-all-co-prtnr-py-PartBadge-Mar2515-1>)

This post is just an introduction to the divide and conquer algorithm. The main algorithms based on this will be discussed in the further posts.

As the name suggests, divide and conquer algorithm first divides the problem and then solves it. So, it is a recursive algorithm which first breaks the problem into sub-problems until the subproblems are simple enough to be solved directly (mainly using brute force) called base case. It is mainly a three-step process. The steps involved are:

Step 1 → Divide the problem into smaller subproblems. For example, if you have an array, then we approach the problem by breaking the array into further smaller sub-arrays.



Breaking an Array Into Subarrays

Step 2 → The second step is to solve the smaller subproblems. For example, if we have to search for an element in an array, then we will now search for the number in the smaller sub-arrays.

Step 3 → The third and the last step is to combine the subproblems to get the solution to the bigger problem. For example, let's consider a sorting problem. If all the sub-arrays are sorted, then it would be easy for us to get sort the main problem by combining the smaller sub-arrays.

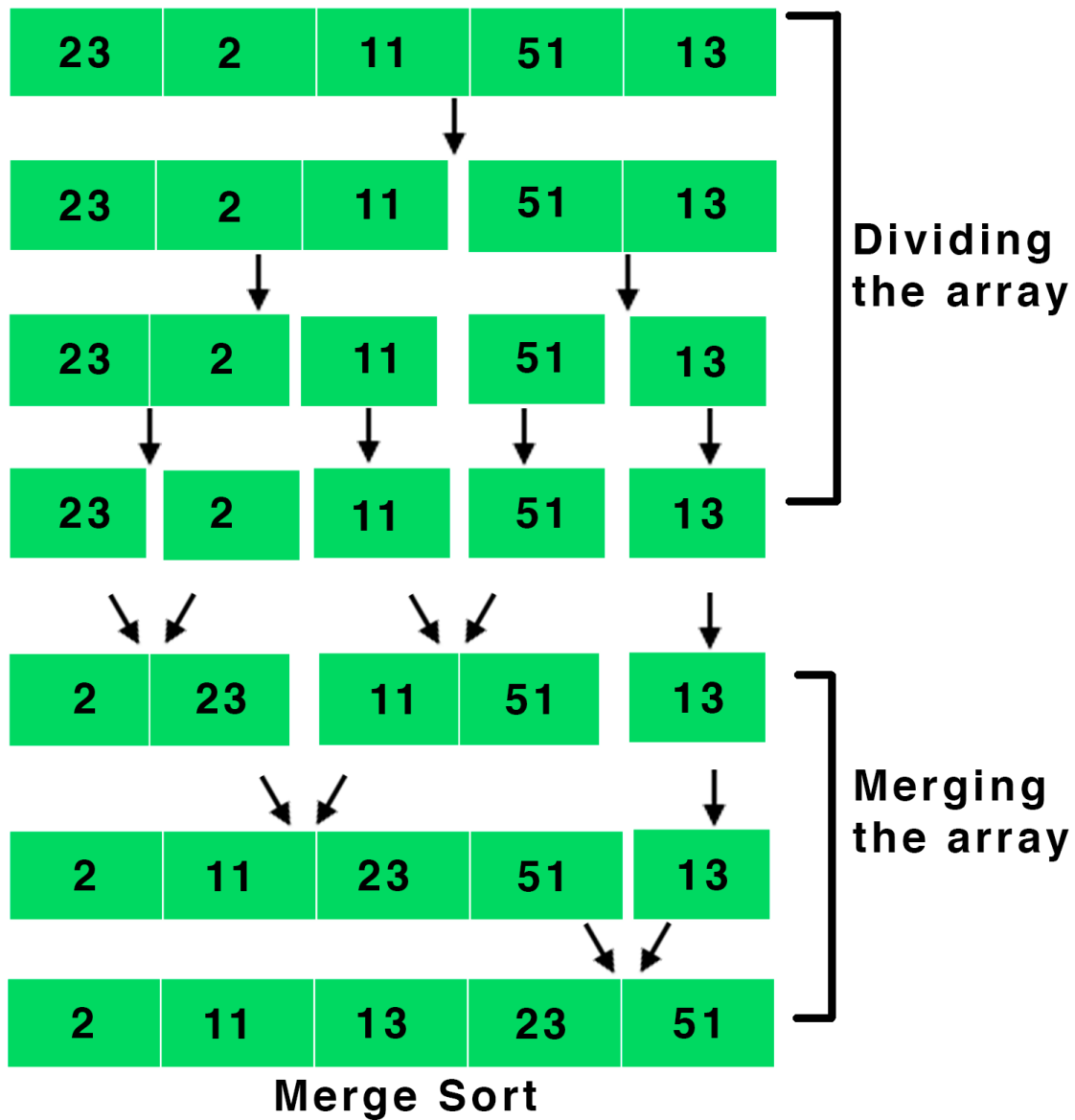
So, we use divide and conquer when the problem is easy to divide into further subproblems and the subproblems are easy to solve.

Use of Divide and Conquer

Many algorithms are based on divide and conquer. I have only mentioned some of them in the list given below. I will discuss each of these in detail in my upcoming posts.

- **Merge Sort** - It is a sorting algorithm which first breaks an array into smaller arrays and then recursively sorts and merges them. The pictorial

representation is given below.



- **Quicksort** - It is a technique in which a pivot is chosen and then all the elements smaller than the pivot are moved on one side of the array and all the elements bigger than the pivot are moved to another side. This is then repeated recursively for the subarrays.



- **Binary Search** - It is an algorithm to search for an element in a sorted array. We first look at the middle element of the array and if the element matches then it is returned, otherwise if it is smaller, then we search in the


left subarray, else in the right subarray.


Search 5




- **Karatsuba Algorithm** - It is an algorithm of fast-multiplying two numbers efficiently. I will discuss the detail in an upcoming post.
- **Closest Pair of Points** - The closest pair of points is a problem in which we are given a set of points in XY plane and we have to find a pair of points with the least distance. The time complexity of the Brute Force solution is $O(n^2)$ i.e., nC_2 but we can solve it in $O(n\log n)$ with divide and conquer. We first sort the points to their x-coordinates and then divide the set into two parts and recursively solve both.

Liked the post?

 (<https://www.facebook.com/sharer/sharer.php?u=https://www.codesdope.com/blog/article/divide-and-conquer/>)

 (<https://twitter.com/intent/tweet?url=https://www.codesdope.com/blog/article/divide-and-conquer/&text=Divide and Conquer &via=codesdope>)

 (<https://plus.google.com/share?url=https://www.codesdope.com/blog/article/divide-and-conquer/>)

 (<https://www.linkedin.com/shareArticle?url=https://www.codesdope.com/blog/article/divide-and-conquer/&title=Divide and Conquer>)

 (https://pinterest.com/pin/create/bookmarklet/?media=https://www.codesdope.com/media/blog_images/1/2018/11/17/fire-and-water-2354583_1280.jpg&url=https://www.codesdope.com/blog/article/divide-and-conquer/&description=Divide and Conquer)