☰ ₱
(/)

🔍 Search tutorials and examples

www.domain-name.com

In this tutorial, you will learn how the divide and conquer algorithm works. We will also compare the divide and conquer approach versus other approaches to solve a recursive problem.

A **divide and conquer algorithm** is a strategy of solving a large problem by

1. breaking the problem into smaller sub-problems

2. solving the sub-problems, and

3. combining them to get the desired output.

To use the divide and conquer algorithm, **recursion** is used. Learn about recursion in different programming languages:

* Recursion in Java (https://www.programiz.com/java-programming/recursion)

* Recursion in Python (https://www.programiz.com/python-programming/recursion)

* Recursion in C++ (https://www.programiz.com/cpp-programming/recursion)

1. **Divide**: Divide the given problem into sub-problems using recursion.

2. **Conquer**: Solve the smaller sub-problems recursively. If the subproblem is small enough, then solve it directly.

3. **Combine:** Combine the solutions of the sub-problems that are part of the recursive process to solve the actual problem.

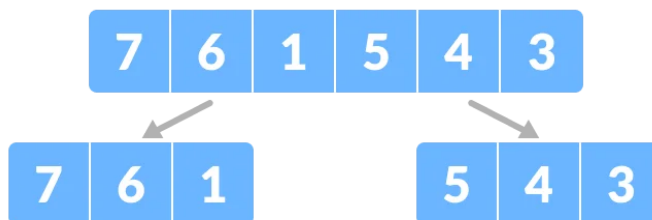Let us understand this concept with the help of an example.

Here, we will sort an array using the divide and conquer approach (ie. merge sort (/dsa/merge-sort)).

1. Let the given array be:
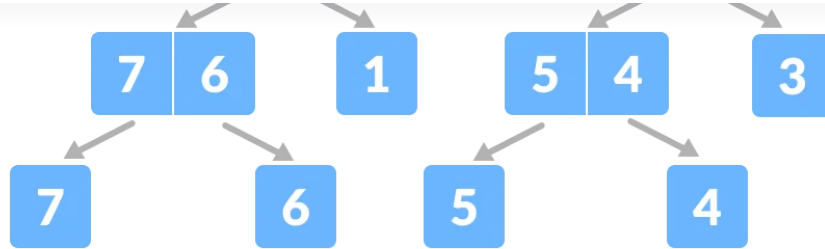

Array for merge sort

2. **Divide** the array into two halves.

Divide the array into two subparts

Again, divide each subpart recursively into two halves until you get individual elements.
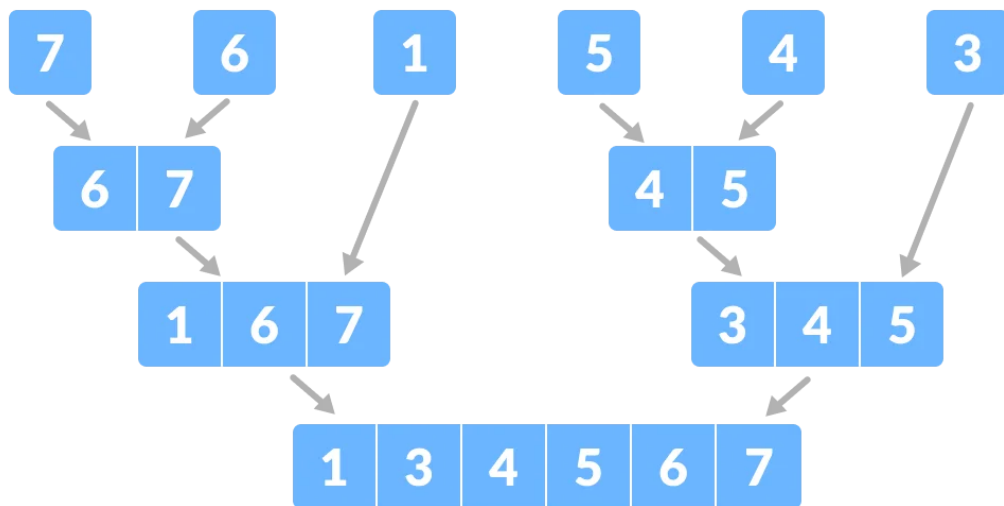
www.domain-name.com

| 7 | 6 | | 1 | | 5 | 4 | | 3 |

| 7 | | 6 | | 5 | | 4 |

Divide the array into smaller subparts

3. Now, combine the individual elements in a sorted manner. Here, **conquer** and **combine** steps go side by side.

| 7 | | 6 | | 1 | | 5 | | 4 | | 3 |

| 6 | 7 | | 4 | 5 |

| 1 | 6 | 7 | | 3 | 4 | 5 |

| 1 | 3 | 4 | 5 | 6 | 7 |

Combine the subparts

Time Complexity

The complexity of the divide and conquer algorithm is calculated using the [master theorem](/dsa/master-theorem) (/dsa/master-theorem).

Let us take an example to find the time complexity of a recursive problem.

For a merge sort, the equation can be written as:

```
T(n) = aT(n/b) + f(n)
     = 2T(n/2) + O(n)
Where,
a = 2 (each time, a problem is divided into 2 subproblems)
n/b = n/2 (size of each sub problem is half of the input)
f(n) = time taken to divide the problem and merging the subproblems
T(n/2) = O(n log n) (To understand this, please refer to the master theorem.)

Now, T(n) = 2T(n log n) + O(n)
          ≈ O(n log n)
```

Divide and Conquer Vs Dynamic approach

Use the divide and conquer approach when the same subproblem is not solved multiple times. Use the dynamic approach when the result of a subproblem is to be used multiple times in the future.

Let us understand this with an example. Suppose we are trying to find the Fibonacci series. Then,

Divide and Conquer approach:

```
fib(n)
    If n < 2, return 1
    Else , return f(n - 1) + f(n -2)
```

Dynamic approach:

```
mem = []
fib(n)
    If n in mem: return mem[n]
    else,
        If n < 2, f = 1
        else , f = f(n - 1) + f(n -2)
        mem[n] = f
        return f
```

In a dynamic approach, `mem` stores the result of each subproblem.

Advantages of Divide and Conquer Algorithm

- The complexity for the multiplication of two matrices using the naive method is `O(n³)`, whereas using the divide and conquer approach (i.e. Strassen's matrix multiplication) is `O(n².8074)`. This approach also simplifies other problems, such as the Tower of Hanoi.

- This approach is suitable for multiprocessing systems.

- [Binary Search](/dsa/binary-search) (/dsa/binary-search)

- [Merge Sort](/dsa/merge-sort) (/dsa/merge-sort)

- [Quick Sort](/dsa/quick-sort) (/dsa/quick-sort)

- Strassen's Matrix multiplication

- Karatsuba Algorithm

Next Tutorial:

Stack   **(/dsa/stack)**

Previous Tutorial:

Master Theorem   **(/dsa/master-theorem)**

Share on:

(https://www.facebook.com/sharer/sharer.php?u=https://www.programiz.com/dsa/divide-and-conquer)

(https://twitter.com/intent/tweet?text= article on Divide and Conquer Algorith &via=programiz&url=https://www.prog and-conquer)

Did you find this article helpful?

😀   🙁