Brief Description of Divide and conquer Algorithm.

Divide and Conquer Algorithm

In divide and conquer algorithm we divide a problem into sub problems and each of the problem is solved seperately.

* The solutions of all these sub problems are finally merged in order to obtain the solution of the original problem.

There are three steps in divide & conquer approach.

1) Divide
2) Conquer
3) Merge.

* In the first step the original problem is divided into sub problems ; the division is done to a point where the sub problems cannot be divided further. This generally follows a recursive approach to divide the problem.

* In the second step, lot of the smaller problems are to be solved. In general these problems solve themselves according to the algorithm.

* In the third step, the sub problems that are solved are recursively combined until they formulate a solution of the original problem. This algorithm works recursively and conquer & merge steps work so closely that they appear as one.

Explanation of Maximum sub Array sum by Divide and conquer:

1) Initially we call find maximum_sub_array (array, low_index, high_index)
2) In the base case the sub array has only a single element.
3) Dividing the array is done on the basis of calculating the mid value. conquering is done by making two recursive calls to find maximum_sub_array and a single call to max cross array (dbg maximum_sum) and the maximum sum of the three is returned.
4) Time complexity is O(n log n) & the recurrence relation is $T(n) = 2 \cdot T(n/2) + O(n)$ where O(n) is the merging time.

Code explanation

* Find the find maximum_sub_array is called in the Python code.
* Here we declare the arr, low_index and the high_index as function parameters.

* Mid value is calculated by using the formula.
mid = high-index+low-index/2.

* Maximum left sub array & maximum right sub array is calculated and is returned using the variables t1, t2 also the indexes of the left sub array (il, jl) and right sub array (ir, jr) are also returned.

* Then maximum-sum is called to find the cross sum assuming the starting of the cross sub array is in the left and ends in the right.

* Finally the maximum sub array is returned along with its sum and indexes.

Example

arr = -1, -2, -3, 4, 5

left index = 3

right index = 4

maximum sub array sum is 9.