DIVIDE AND CONQUER APPROACH

MAX SUB-ARRAY PROBLEM

Given an array **arr[]** of size **N**. The task is to find the sum of the contiguous subarray within a **arr[]** with the largest sum.

Algorithm

1) Divide the given array in two halves

2) Return the maximum of following three

    a) Recursively calculate the Maximum subarray sum in left half

    b) Recursively calculate the Maximum subarray sum in right half

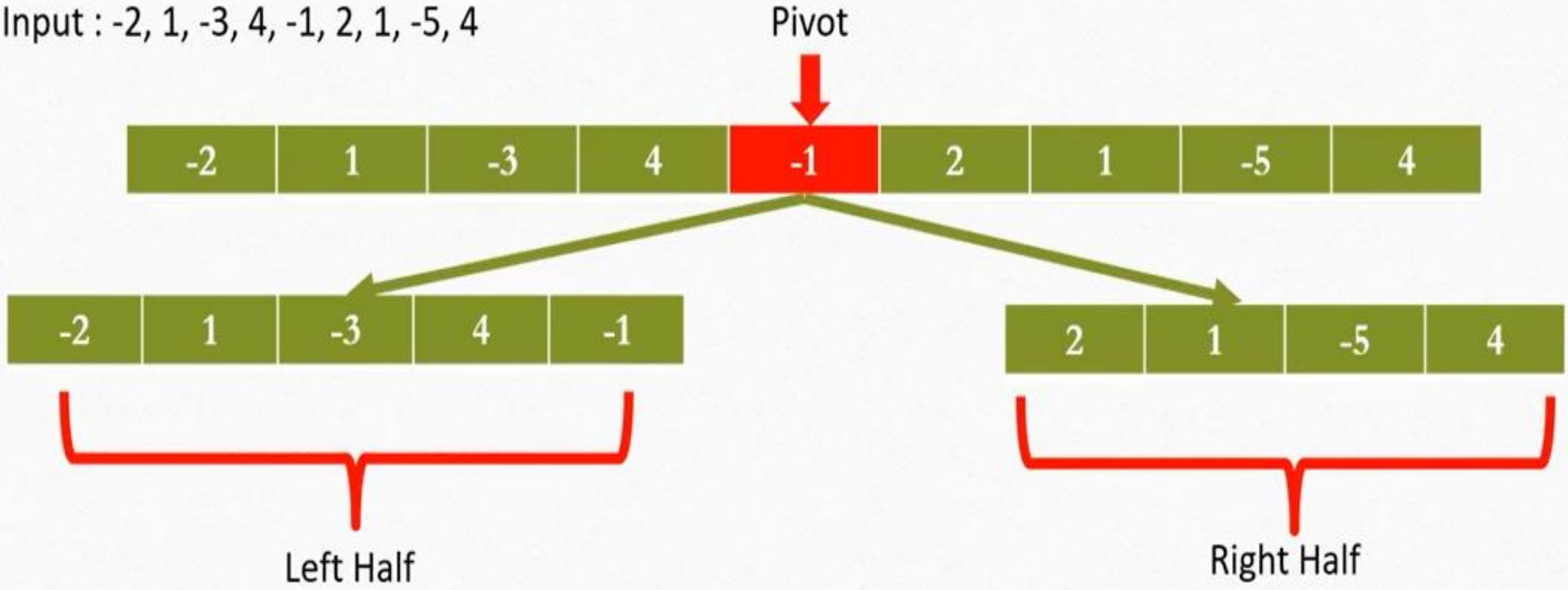
    c) Recursively calculate the Maximum subarray sum such that the subarray crosses the midpoint.

        i. Find the maximum sum starting from mid point and ending at some point on left of mid.

        ii. Find the maximum sum starting from mid + 1 and ending with some point on right of mid + 1.

        iii. Finally, combine the two and return.

Input : -2, 1, -3, 4, -1, 2, 1, -5, 4

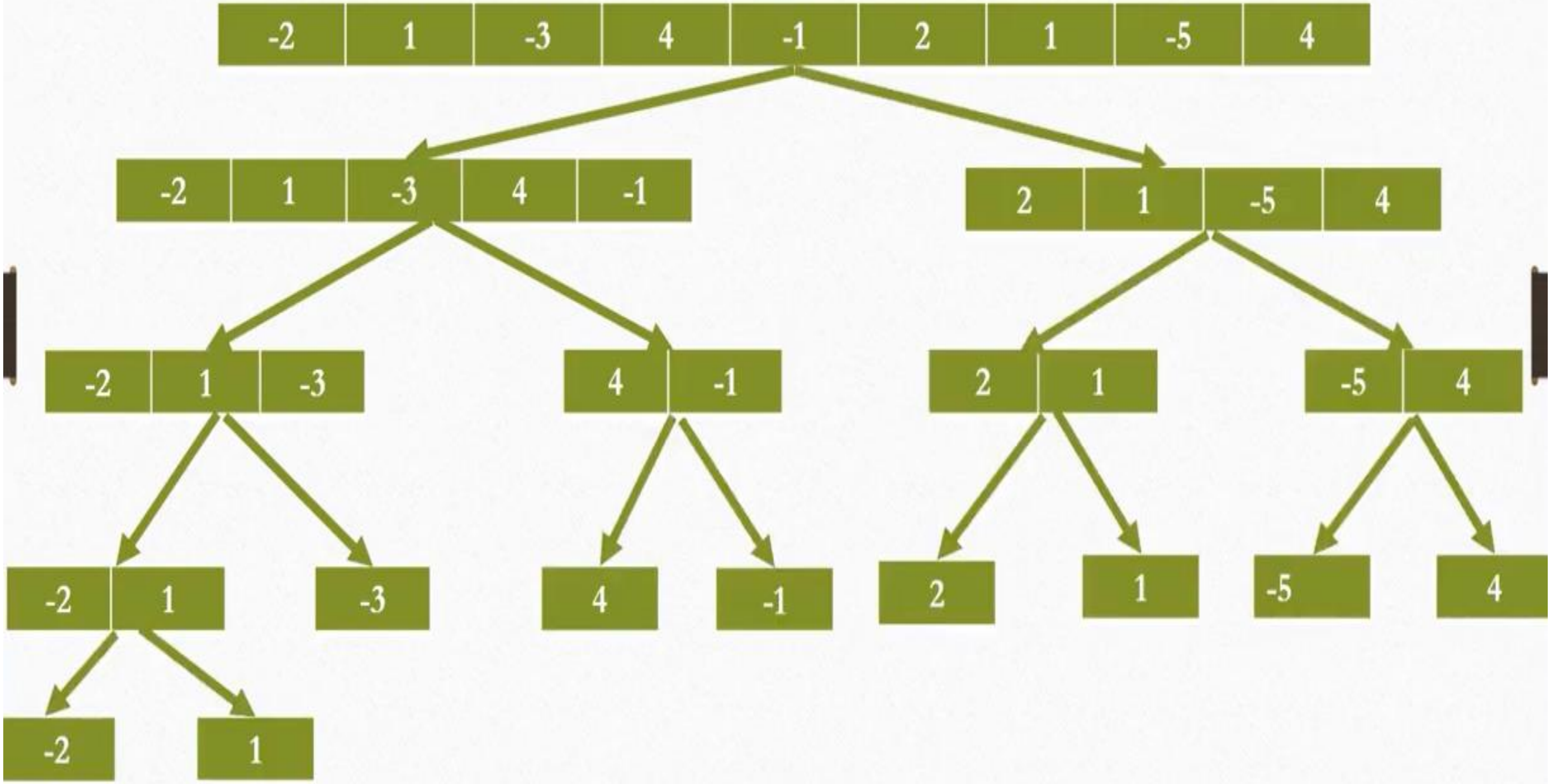


$$\textbf{Index of Pivot} = \frac{\textbf{(begin + end)}}{2} = \frac{\textbf{(0+8)}}{2} = \textbf{4}$$

Left Half = begin to pivot (inclusive)

Right Half = pivot +1 to end

Input : -2, 1, -3, 4, -1, 2, 1, -5, 4



-2 | 1 | -3 | 4 | -1 | 2 | 1 | -5 | 4

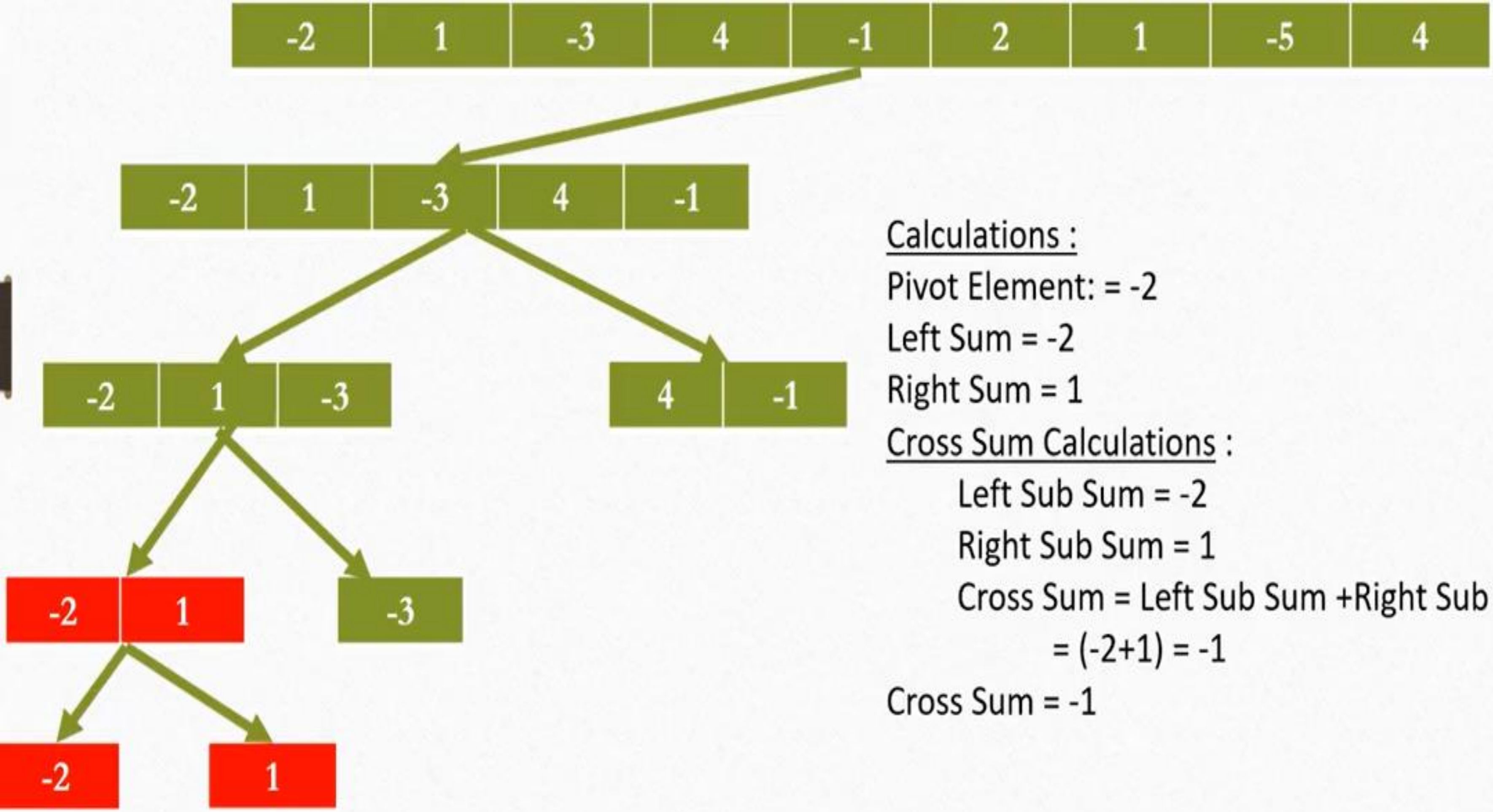
-2 | 1 | -3 | 4 | -1   2 | 1 | -5 | 4

-2 | 1 | -3   4 | -1   2 | 1   -5 | 4

-2 | 1   -3   4   -1   2   1   -5   4

-2   1

Input : -2, 1, -3, 4, -1, 2, 1, -5, 4



Calculations :

Pivot Element: = -2

Left Sum = -2

Right Sum = 1

Cross Sum Calculations :

Left Sub Sum = -2

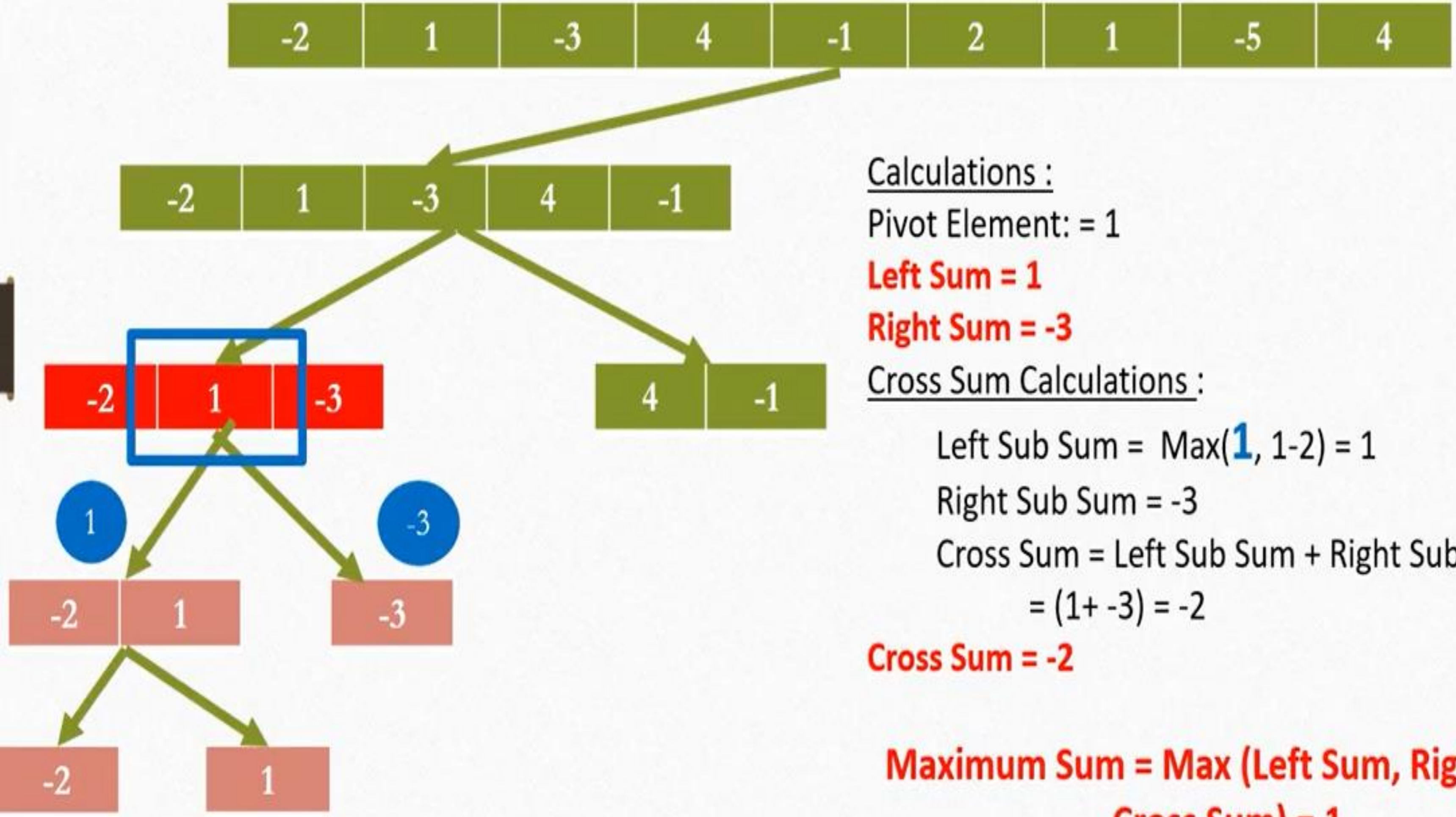
Right Sub Sum = 1

Cross Sum = Left Sub Sum +Right Sub Sum

= (-2+1) = -1

Cross Sum = -1

Input : -2, 1, -3, 4, -1, 2, 1, -5, 4



Calculations :

Pivot Element: = 1

Left Sum = 1

Right Sum = -3

Cross Sum Calculations :

Left Sub Sum = Max(**1**, 1-2) = 1

Right Sub Sum = -3

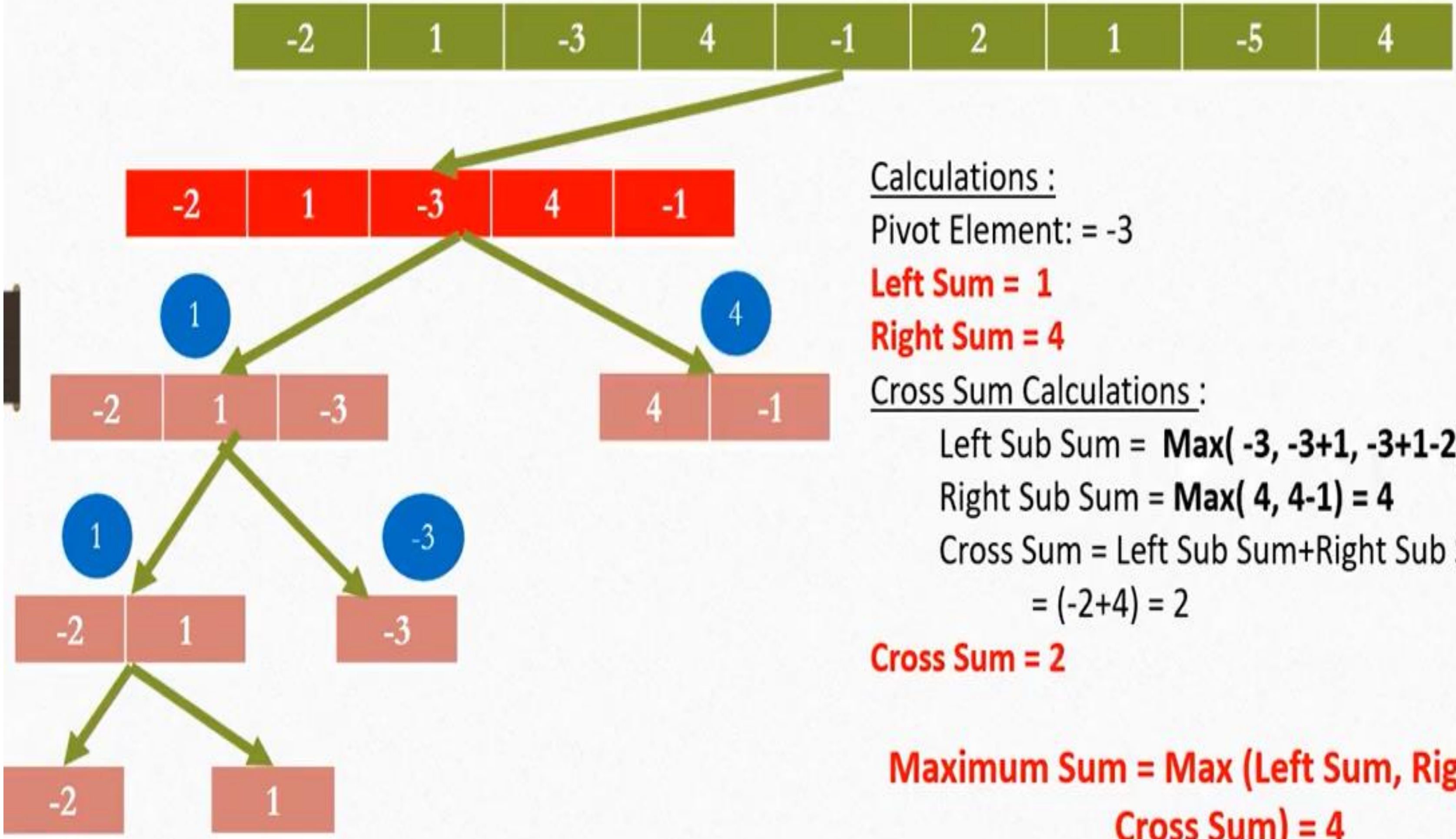
Cross Sum = Left Sub Sum + Right Sub Sum

= (1+ -3) = -2

Cross Sum = -2

Maximum Sum = Max (Left Sum, Right Sum Cross Sum) = 1

Input : -2, 1, -3, 4, -1, 2, 1, -5, 4



Calculations :

Pivot Element: = -3

Left Sum = 1

Right Sum = 4

Cross Sum Calculations :

Left Sub Sum = **Max(-3, -3+1, -3+1-2) = -2**

Right Sub Sum = **Max(4, 4-1) = 4**

Cross Sum = Left Sub Sum+Right Sub Sum

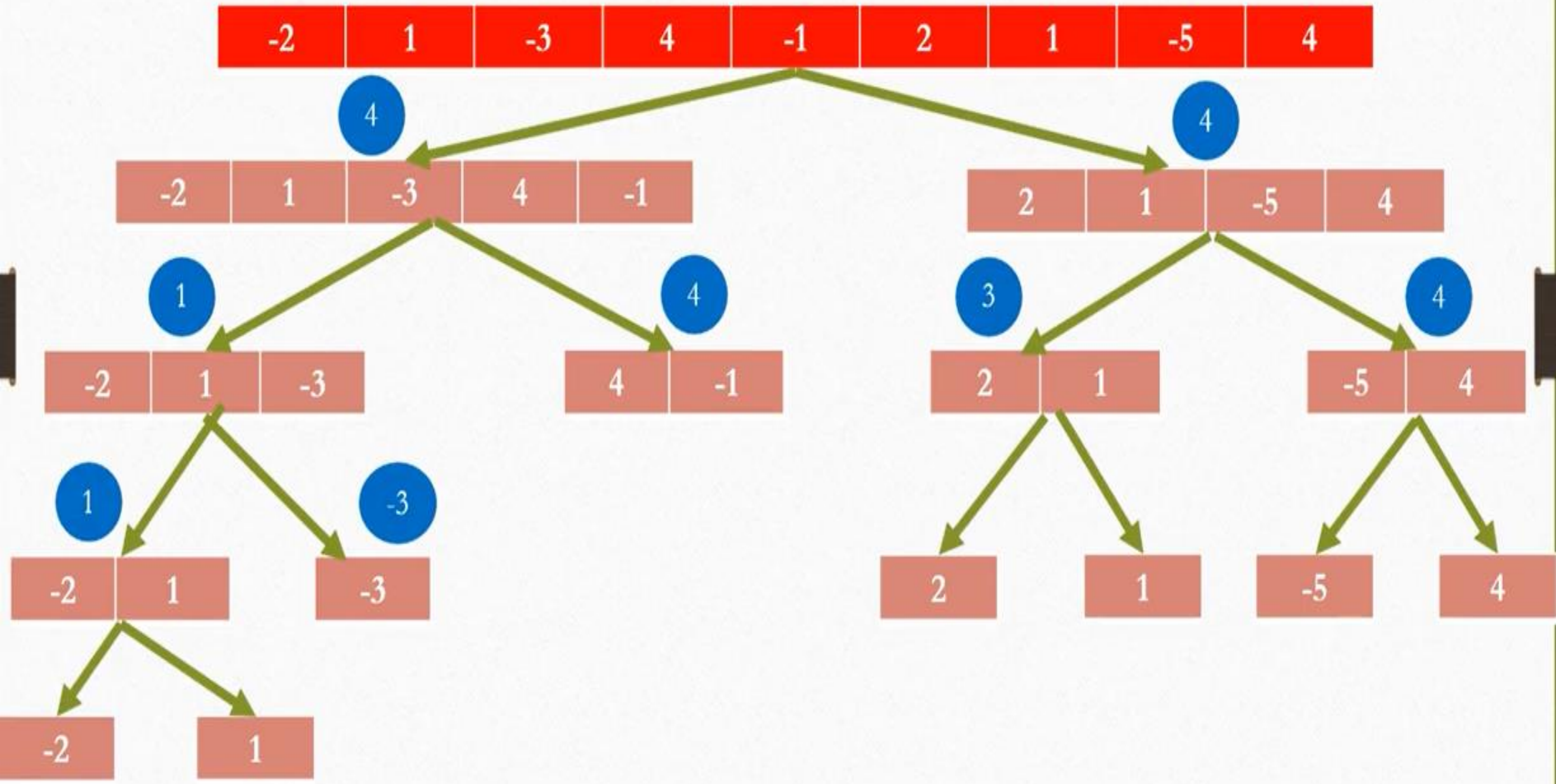
= (-2+4) = 2

Cross Sum = 2

Maximum Sum = Max (Left Sum, Right Sum Cross Sum) = 4

Input : -2, 1, -3, 4, -1, 2, 1, -5, 4

Left Sum =4, Right Sum =4,
Cross Sum = ??



Left Sum =4, Right Sum =4,
Cross Sum = ??

Input : -2, 1, -3, 4, -1, 2, 1, -5, 4

-2	1	-3	4	-1	2	1	-5	4

Cross Sum Calculation :

Pivot Element: -1

Left Sub Sum = Max (-1, -1+4, -1+4-3, -1+4-3+1, -1+4-3+1-2)
= Max (-1, 3, 0, 1, -2)
= 3

Right Sub Sum = Max (2, 2+1, 2+1-5, 2+1-5+4)
= Max (2, 3, -2, 2)
= 3

Cross Sum = Left Sub Sum + Right Sub Sum = (3+3) = 6

Cross Sum = 6

Maximum Sum = Max (Left Sum, Right Sum, Cross Sum) = 6

```
FIND-MAXIMUM-SUBARRAY($A$, $low$, $high$)
if $high == low$
    return ($low$, $high$, $A[low]$)                // base case: only one element
else $mid = \lfloor (low + high)/2 \rfloor$
    ($left\text{-}low$, $left\text{-}high$, $left\text{-}sum$) =
        FIND-MAXIMUM-SUBARRAY($A$, $low$, $mid$)
    ($right\text{-}low$, $right\text{-}high$, $right\text{-}sum$) =
        FIND-MAXIMUM-SUBARRAY($A$, $mid + 1$, $high$)
    ($cross\text{-}low$, $cross\text{-}high$, $cross\text{-}sum$) =
        FIND-MAX-CROSSING-SUBARRAY($A$, $low$, $mid$, $high$)
    if $left\text{-}sum \geq right\text{-}sum$ and $left\text{-}sum \geq cross\text{-}sum$
        return ($left\text{-}low$, $left\text{-}high$, $left\text{-}sum$)
    elseif $right\text{-}sum \geq left\text{-}sum$ and $right\text{-}sum \geq cross\text{-}sum$
        return ($right\text{-}low$, $right\text{-}high$, $right\text{-}sum$)
    else return ($cross\text{-}low$, $cross\text{-}high$, $cross\text{-}sum$)
```

The running time T(n), recurrence relation of finding maximum sub-array is:

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1\ , \\ 2T(n/2) + \Theta(n) & \text{if } n > 1\ . \end{cases}$$