Divide and Conquer

Suman Mukherjee

July 23, 2023

1 Introduction

Divide and conquer is a problem-solving approach that breaks a problem down into smaller subproblems, solves each subproblem recursively, and then combines the solutions to the subproblems to solve the original problem. This approach is often used to solve problems that can be expressed in terms of a recursive relationship.

2 Algorithm

Merge Sort is a Divide and Conquer algorithm. It divides the input array into two halves, calls itself for the two halves, and then it merges the two sorted halves.

Algorithm 1 MergeSort(low,high)

Require: array[n],low,high
Ensure: $n \geq 1$
  if $low \leq$high **then** ▷ If there are more than 1 element
    $mid \leftarrow \frac{low+high}{2}$ ▷ Finds where to split
    MergeSort(low,mid) ▷ Solve the solve problems
    MergeSort($mid+1$,high)
    Merge(low,mid,high) ▷ Combine the solutions
  end if

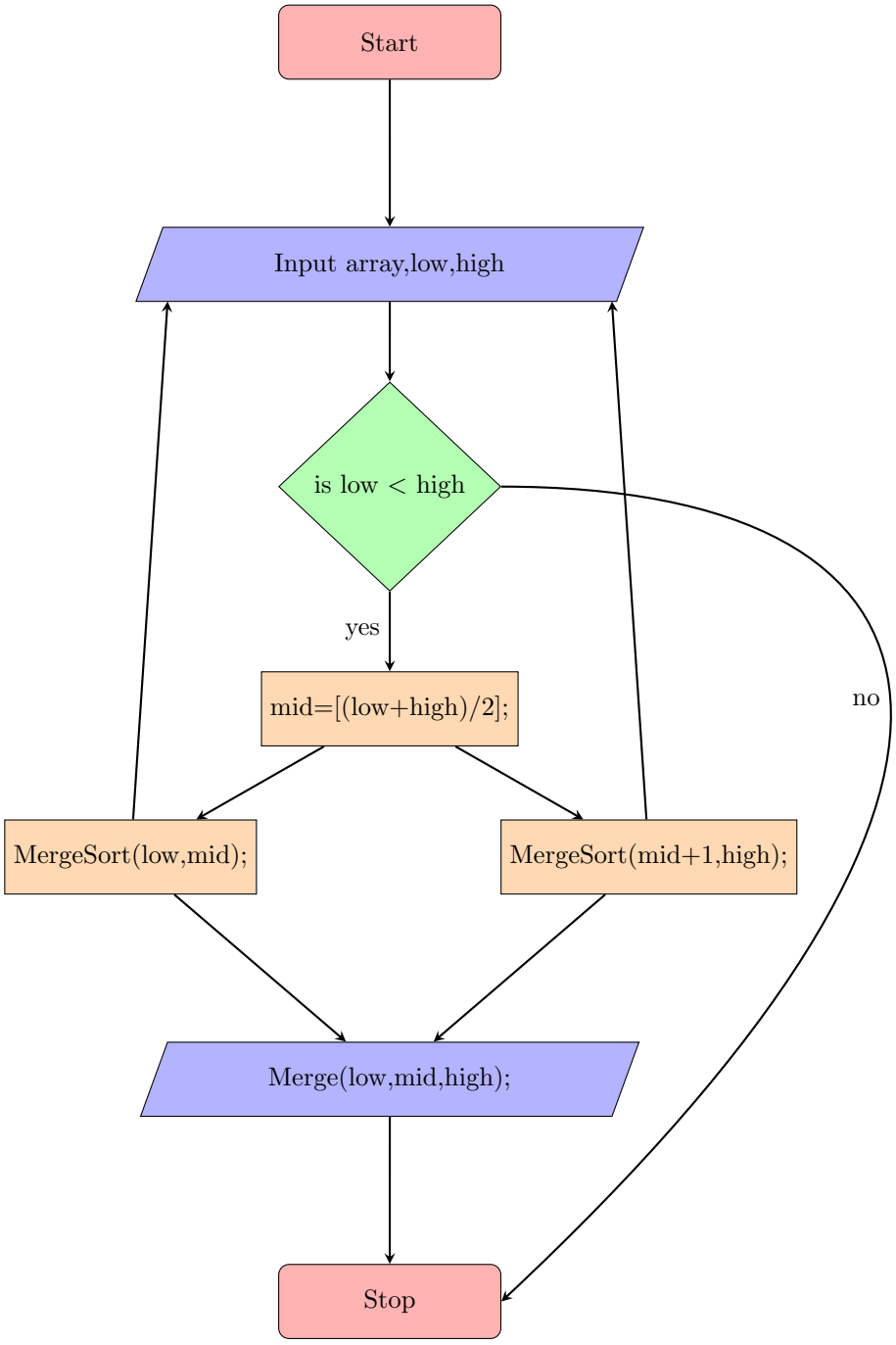
The merge() function is used for merging two halves. The merge(low, mid, high) is a key process that assumes that arr[l..mid] and arr[mid+1..high] are sorted and merges the two sorted sub-arrays into one

Algorithm 2 Merge(low,mid,high)

Require: array[low:mid],array[mid+1,high],b[],low,high,mid
  $h \leftarrow low$
  $i \leftarrow low$
  $j \leftarrow mid + 1$
  while ($h \leq$mid) and ($j \leq high$) **do**
    if (array[h] $\leq array[j]$) **then**
      b[i] $\leftarrow array[h]$
      h$\leftarrow$ (h+1)
    else
      b[i] $\leftarrow array[j]$
      j$\leftarrow$ (j+1)
    end if
    i$\leftarrow$ (i+1)
  end while
  if (h$\geq mid$) **then**
    for (k$\leftarrow$ j to $high$) **do**
      b[i] $\leftarrow array[k]$
      i$\leftarrow$ (i+1)
    end for
  else
    for (k$\leftarrow$ h to mid) **do**
      b[i] $\leftarrow array[k]$
      i$\leftarrow$ (i+1)
    end for
  end if
  for (k$\leftarrow$ low to $high$) **do**
    array[i] $\leftarrow b[k]$
  end for

3 Flowchart

The following is a flowchart for the divide and conquer approach to sort in a list of numbers:firstly the array divides itself recursively into sub-arrays until the base case is reached.Then the sub-arrays are sorted using recursion. Lastly in this step makes use of the merge( ) function to combine the sub-arrays into the final sorted array..



4 Example Program

The following Python program sorts a list of numbers using the divide and conquer approach.

```
def merge_sort(array):
  if len(array) <= 1:
    return array

  middle = len(array) // 2
  left = merge_sort(array[:middle])
  right = merge_sort(array[middle:])

  return merge(left, right)

def merge(left, right):
  merged = []
  i = 0
  j = 0

  while i < len(left) and j < len(right):
    if left[i] <= right[j]:
      merged.append(left[i])
      i += 1
    else:
      merged.append(right[j])
      j += 1

  merged += left[i:]
  merged += right[j:]

  return merged

if __name__ == "__main__":
  array = [10, 5, 2, 1, 8, 7, 6, 3, 4, 9]
  sorted_array = merge_sort(array)
  print(sorted_array)
```

5 Conclusion

Divide and conquer is a powerful problem-solving approach that can be used to solve a wide variety of problems. This approach is often used to solve problems that can be expressed in terms of a recursive relationship. The divide and conquer approach can be used to solve problems more efficiently than other approaches, such as brute force.