Roll NO : 33231

classmate
Date
Page

ASSIGNMENT 1

Problem Statement : Write a program to find Maximum and Minimum element in an array using Divide and Conquer strategy and verify the time complexity.

Theory :

1) Write pseudocode to find minimum and maximum numbers from the given list of elements using straight forward method. Also calculate number of comparisons required.

→
```
void findMinMax (int a[], int size) {
    int min, max;
    min = max = a[0];
    for (int i=1; i<size; i++) {
        if (a[i] > max)
            max = a[i]
        else if (a[i] < min)
            min = a[i];
    }
    cout << min << max;
}
```

No of comparisons = $2n$

where n = size of array.

2) Write control abstraction using Divide and Conquer strategy.

→ A control abstraction is a procedure whose flow of control is clear but whose primary operations are specified by other procedures

whose precise meanings are left undefined. The control abstraction for divide and conquer technique is DANDC(P), where P is the problem to be solved.

```
DANDC(P) {
    if SMALL(P) then return S(P);
    else {
        divide p into smaller instances p1, p2, .... pk
        apply DANDC to each of these sub problems;
        return (COMBINE(DANDC(p1),
        DANDC(p2), ...., DANDC(pk)));
    }
}
```

SMALL(P) is a boolean valued function which determines whether the input size is small enough so that the answer can be computed without splitting. If this is so function 'S' is invoked otherwise, the problem 'p' into smaller sub problems. These sub-problems p1, p2, ... pk are solved by recursive application of DANDC.

If the sizes of the 2 sub problems are approximately equal then the computing time of DANDC is:

$$T(n) = \begin{cases} g(n) & n \text{ small} \\ 2T(n/2) + f(n) & \text{otherwise} \end{cases}$$

where $T(n)$ = time for DANDC on 'n' inputs

$g(n)$ = time to complete answer directly (for small i/p)

$f(n)$ = time for Divide and Conquer

35231

classmate
Date
Page

3) How Divide and Conquer strategy is used to find minimum and maximum numbers from the given list of elements? Explain with example.

→ Consider the given list of elements as

7 65 23 1 45

Initially, Beg = 0 , End = 4

Now, the array is divided into two halves (left half)

∴ Beg = 0 , End = 2

Again, array is divided (left half)

∴ Beg = 0 , End = 1

Now, we have two elements in array. Hence we compare them to find min and max

∴ min = 7 , max = 65

Going back to previous recursive call (right half)

Beg = 2 , End = 2

∴ min = 23 , max = 23

Taking the whole left sub-array

Beg = 0 , End = 2

∴ min = 7 , max = 65

Dividing the array again (right half)

∴ Beg = 3 , End = 4

Now, we have two elements in array. Hence we compare them to find min & max.

∴ min = 1 , max = 45

Now taking the whole array; both left and right subarrays

∴ BEG = 0 , END = 4

∴ min = 1 , max = 65

35231

4) Write pseudocode of MinMax problem using Divide and Conquer.

→ Pair MinMax (array, array-size)
  if array-size = 1
    return element as both max and min
  else if array-size = 2
    one comparison to determine max & min
    return that pair
  else
    recur for max & min of left half
    recur for max & min of right half
    one comparison determines true max
    one comparison determines true min
    return pair of max and min.

5) "Divide and Conquer method is taking a smaller number of comparisons than straight forward method". Justify the statement.

→ The recurrence relation for min max problem using divide and conquer :

$$T(n) = \begin{cases} T(n/2) + T(n/2) + 2 & n > 2 \\ 1 & n = 2 \\ 0 & n = 1 \end{cases}$$

When n is a power of two, $n = 2^k$ for some positive integer k, then

$$T(n) = 2T(n/2) + 2$$
$$= 2(2T(n/4) + 2) + 2$$
$$= 4T(n/4) + 4 + 2$$
$$\vdots$$

33231

$$= 2^{k-1} T(2) + \sum_{i=1}^{k-1} 2^i$$

$$= 2^{k-1} + 2^k - 2$$

$$= \frac{3n}{2} - 2$$

Note that $3n/2 - 2$ is the best, average and worst case number of comparisons when n is a power of two.
Compared with the straight forward method $(2n - 2)$ this method saves 25% in comparison.

6) Explain time analysis to divide and conquer solution of min max problem.

→ Recurrence Relation :

$$T(n) = \begin{cases} T(n/2) + T(n/2) + 2 & n > 2 \\ 1 & n = 2 \\ 0 & n = 1 \end{cases}$$

For $n = 2^k$, for some integer k

$$T(n) = 2T(n/2) + 2$$
$$= 2(2T(n/4) + 2) + 2$$
$$= 4T(n/4) + 4 + 2$$
$$\vdots$$
$$= \frac{3n}{2} - 2$$

Asymptotic notation = $O(n)$

Conclusion : I have successfully implemented the program to find minimum and maximum element in an array using divide & conquer