Find Local Minimum in Grid N\*N Matrix

**Abstract:**

The local minimum in a graph is a node whose value is less than its adjacent nodes. Generally, we can find a local minimum of N\*N grid matrix in O (N^2) time complexity using Brute Force technique. By using the Divide and Conquer technique we can optimize the time complexity of this problem. In this technique, we divide the matrix recursively and then solve the divided problem in O (N) worst time complexity.

1 Problem Statement

Suppose you are given an n \* n grid graph G. (An n \* n grid graph is just the adjacency graph of n \* n chessboard. To be completely precise, it is a graph whose node set is the set of all ordered pairs of natural numbers (i, j), where 1 ≤ i ≤n and 1 ≤ j ≤n; the nodes (i, j) and (k, l) are joined by an edge if and only if |i-k| + |j-l| = 1.) Each node v is labeled by a real number xv; you may assume that all these labels are distinct. A node v of T is a local minimum if the label xv is less that the label xw for all nodes w that are joined to v by an edge. Show how to find a local minimum of G using only O(n) probes to the nodes of G. (Note that G has n2 nodes.)

2 The conventional approach

In the conventional approach, we consider the first node and then compare its value with its adjacent nodes. If its value is less than its all adjacent nodes then it is the local minimum in the graph and we will stop there. Otherwise, we will consider the next node in the same row and the same procedure is repeated for comparing that node with their all adjacent nodes until we find the correct solution. In this way, the same procedure is repeated for all nodes until we find the local minimum. In the worst case, it will take O (N^2) time complexity to find the correct local minimum in the graph of N\*N nodes.

2.1 Analysis

Recurrence Relation for the Conventional approach

T (n2) = T (n2 – 1) + c --------------- (1)

T (1) = c

Put n2 = m in (1) we get,

T (m) = T (m – 1) + c ---------------------- (2)

T (m – 1) = T (m – 2) + c put in (2)

T (m) = T (m – 2) + c + c -------------------- (3)

T (m – 2) = T (m – 3) + c put in (3)

T (m) = T (m – 3) + c + c + c

T (m) = T (m – 3) + 3c

.

.

T (m) = T (m – k) + k\*c

m – k =1 therefore k = m – 1

T (m) = T (1) + (m- 1)\*c

T (m) = c + (m-1) \* c

T (m) = m \*c

T (m) = O (m) but m=n2

T (n2 ) = O (n2 )

**3 Divide and conquer approach**

3.1 Divide and conquer Strategy

As conventional approach takes O (n2) time complexity in worst case so we can optimize it to O (n) time complexity by using Divide and Conquer technique. In this strategy, we find the minimum element in the middle row and then check whether it is a local minimum. If yes then stop else we check for an adjacent minimum element in the same column. Consider the new minimum element and divide the n\*n grid into n2/4 grid such that the new minimum element is present in that quadrant. Repeat the above procedure again to find a local minimum in that quadrant.

3.2 Algorithm

1. Find the middle row.
2. Find minimum in that middle row
3. Check if it is local minimum If yes return and stop
4. Else find a minimum adjacent element in the same column
5. Now, consider the quadrant having that new minimum element
6. Go to step 1 to find a middle row of the new quadrant.

3.3 Proof

Consider n\*n grid nodes represented as matrix ‘A’:

In the Divide and Conquer strategy, we have to find a middle row say it is n/2th row. So, in a n/2th row, we will find the minimum element So it requires ‘**n’** comparison. We will check for the local minimum condition if true then stop otherwise check the adjacent minimum element in the same column and consider it as a new minimum element.

Now we will divide the matrix as following:

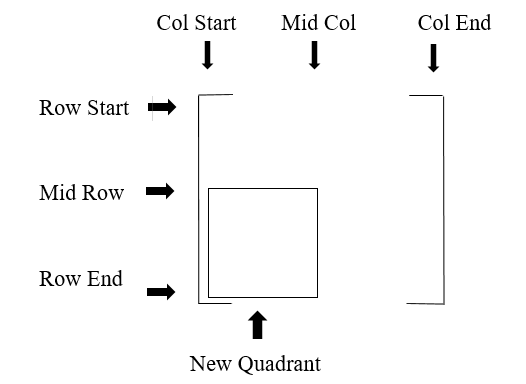
If new minimum(j) < midColumn(j) then columnEnd = midColumn

Else columnStart = midColumn

If new minimum (i) < midRow(i) then rowEnd = midRow

Else rowStart = midRow

By above conditions we will get the new quadrant and further operations will be performed in it.



Again, we will find out the middle row in the new quadrant and then the minimum element in that row.

So, it almost requires ‘**n/2’** comparisons. If the further division is done then it requires **‘n/4’** comparisons.

In this way, we are doing the comparison as follow:

n, n/2, n/4, n/8, …, 1

Total comparisons = n + + + + + … +

We have = 1,

So, we get k = log n

Therefore,

* n \*
* n \*
* n \*
* 2n\*
* 2n –
* 2n –
* 2n – 2
* 2(n-1)

Therefore, O (n)

**3.4 Analysis**

T (n2) =T (n2/4) +n

T (1) = O (1)

Put n2=m

T (m) =T (m/4) + ---------------------------------- (1)

T (m/4) = T (m/42 ) +

T (m/4) = T (m/42 ) +

T (m) = T (m/42 ) + + --------------------- (2)

T (m/42 ) = T (m/43 ) + 2)

T (m/42 ) = T (m/43 ) + 2

T (m) = T (m/43 ) + 2 + + --------- (3)

….

..

T (m) = T (m/4k) + k-1 + k-2 + …. + 2 + 1 +/20

m/4k = 1 therefore k =

T (m) = T (1) +

T (m) = T (1) +

T (m) = T (1) +

T (m) = T (1) +

Put k =

T (m) = T (1) +

T (m) = O (1) + – 2

T (m) ≈ O ()

T (n2) =O (n) since m =n2

**4 Performance evaluation**

|  |  |  |  |
| --- | --- | --- | --- |
| **No of Inputs (n)** |  | **Brute Force** | **Divide & Conquer** |
| **5** | Time in Microsecond | 7.43 | 7.44 |
| No. Of comparisons | 25 | 12 |
| **10** | Time in Microsecond | 7.45 | 7.46 |
| No. Of comparisons | 100 | 27 |
| **25** | Time in Microsecond | 7.50 | 7.51 |
| No. Of comparisons | 625 | 56 |
| **50** | Time in Microsecond | 7.52 | 7.53 |
| No. Of comparisons | 2500 | 110 |
| **100** | Time in Microsecond | 7.54 | 7.55 |
| No. Of comparisons | 10000 | 211 |
| **500** | Time in Microsecond | 7.57 | 7.56 |
| No. Of comparisons | 250000 | 1012 |
| **1000** | Time in Microsecond | 7.60 | 7.58 |
| No. Of comparisons | 1000000 | 2013 |

So from the above, it is clear that for a large value of n (input) Divide and Conquer technique requires less number of comparisons and takes less amount of time than the conventional brute force approach. Divide and Conquer requires an approximately 2n number of comparisons on the other hand brute force approach require n2 comparison. Hence it is clear that the Divide and conquer technique is efficient than the conventional approach.