Khaithik .S. Anand

IBM18CS046

Number of Islands ⇒ Disjoint Sets

Algo:

1) Start
2) int result = 0
3) Traverse through matrix
    1) if arr[i][j] = 1
        ⇒ check its neighbours
        ⇒ if neighbour = 1, take union of it and its neighbours
4) Create array to store frequency of sets
5) Traverse matrix again, if arr[i][j] ⇒ find its set
6) If its frequency is 0 result++

Function: Pseudo Code

```
int countIsland (vector <vector<int>> a)
{
    int n = size (a)
    int m = size (a[0])
    dis => disjointunionset (n * m)
    for i in (0 -> n):
        for k in (0 -> m):
```

Khanthish S. Anand

IBM Bangalore

```
if a[j][k] = 0  continue;
if j+1<n & a[j+1][k] == 1)
dis.union (j+m+k, (j+1)+m+k)
if (j-1 >= 0 & a[j-1][k] == 1)
dis.union (j+m+k, j+m+k+1)
if (k+1<m & a[j][k+1] == 1)
dis.union (j+m+k, j+m+k+1)
if (j+1<n & k+1<m & a[j+1][k+1] = 1)
dis.union (j+m+k, (j+1)+m+k+1)
if (j-1 >= 0 & k-1 >= 0 & a[j-1][k-1] == 1)
dis.unio (j+m+k, (j-1)+m+k-1)
```

Disjoint Union Sets:

```
vector<int> rank, parent
int n;
DisjointUnionSets(int n)
{
    rank.resize(n);
    parent.resize(n);
    this->n = n;
    iota(begin(parent), end(parent), 0);
}
int find(int x) {
    if (parent[x] != x)
        return find(parent[x]);
    return x;
}
```