Student: Dhiraj Bag, Roll: 001911001033 – AI Lab Assignment 3 – Point[5]:

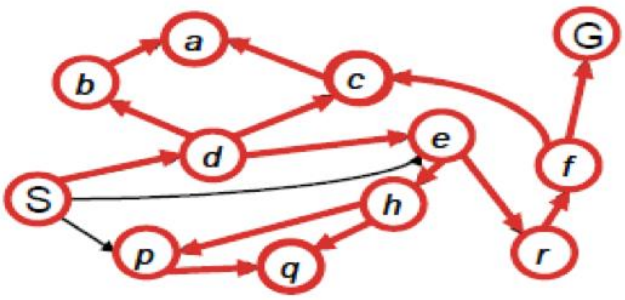

Figure: Given Input Graph

Iterative Deepening Algorithm / Iterative Deepening Depth First Search (IDDFS) – Observations :

In iterative deepening, we incrementally change the allowed path depth from 1 to max_depth (5 in our case). Due to which, the initial iterations hover near and around the source vertex (start vertex / root for tree). As a result, the vertices close to the source vertex are visited first, irrespective of any existence of long/high-depth path.

As an example, executing IDDFS on the input graph gives:

```
IDDFS from S with depth 1:
:
: --> S --> d --#
      :
      : --> e --#
      :
      : --> p --#
```

Notice that node ‘p’ is visited in the very first deepening iteration with depth = 1. ‘p’ is visited just after visiting ‘S’, ‘d’ and ‘e’.

But in normal Depth First Search with a maximum depth, we do NOT start with allowed path depth = 1, RATHER, we allow every path that has depth <= max_depth (5 in our case). Due to which DFS traversal visits and extends the current path as long as path depth is <= max_depth, irrespective of whatever is close to the source vertex. As a result, some vertices may get visited very late although they are very close to the source vertex.

As an example, executing DFS on the input graph gives:

```
DFS from S with max depth 5
:
: --> S --> d --> b --> a --#
            :
            : --> c --#
            :
            : --> e --> h --> p --> q --#
                  :
                  : --> r --> f --> G --#
```

Notice that node ‘p’, in spite of being adjacent to source ‘S’, gets visited very late. First the nodes ‘S’, ‘d’, ‘b’, ‘a’, ‘c’, ‘e’ and ‘h’ are visited. After them ‘p’ is visited.

Also note that if we had run a Breadth First Search from source S, the closest vertices would have been visited first. But the problem with BFS is its worst-case space complexity. It has O(n) space complexity, n being the number of vertices.

Iterative Deepening Search or IDDFS works like BFS but has lower worst-case space complexity than BFS. In fact, its worst-case space complexity is same as that of DFS, i.e., O(max_depth).