Breadth-First Search

•Like DFS, a Breadth-First Search (BFS) traverses a connected component of a graph, and in doing so definesaspanningtreewithseveralusefulproperties

-The starting vertex *s* has level 0, and, as in DFS, defines that point as an “anchor.”

-In the first round, the string is unrolled the length of one edge, and all of the edges that are only one edge away from the anchor are visited.

-These edges are placed into level 1

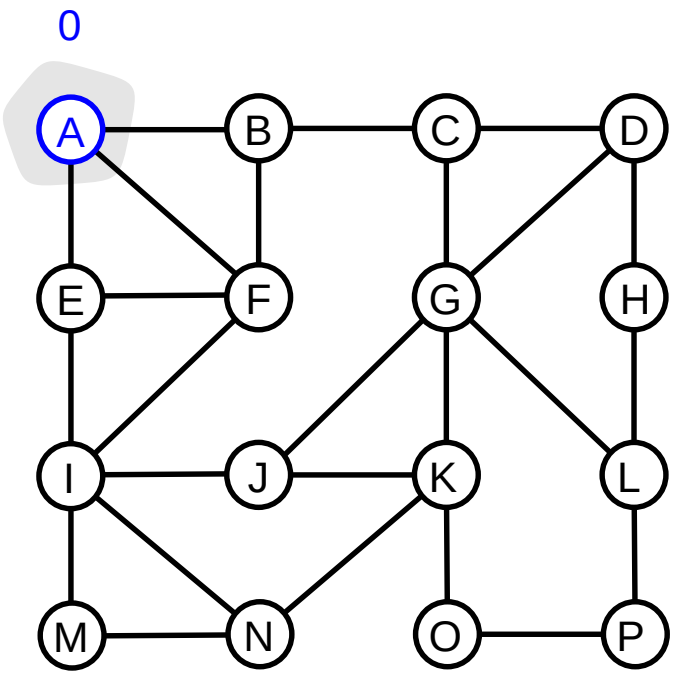
-In the second round, all the new edges that can be reached by unrolling the string 2 edges are visited and placed in level 2.

-This continues until every vertex has been assigned a level.

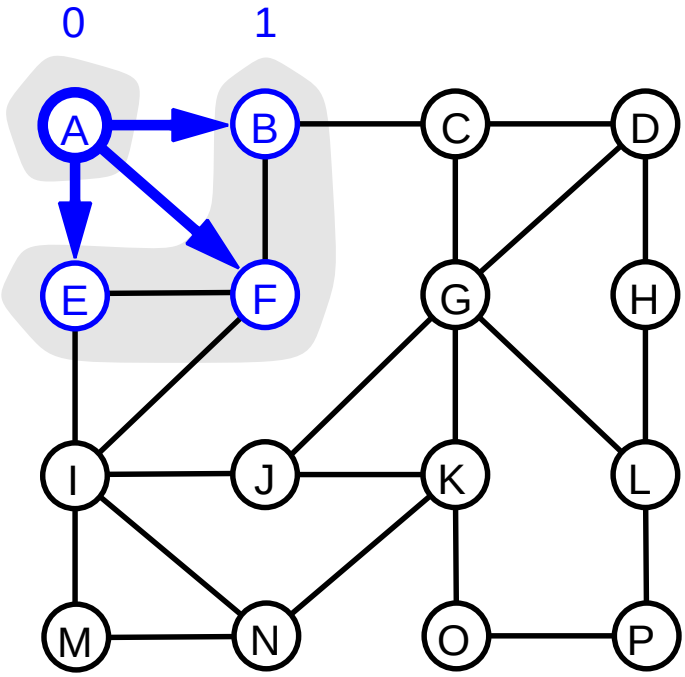
-The label of any vertex *v* corresponds to the length of the shortest path from *s* to *v*.

BFS - A Graphical Representation

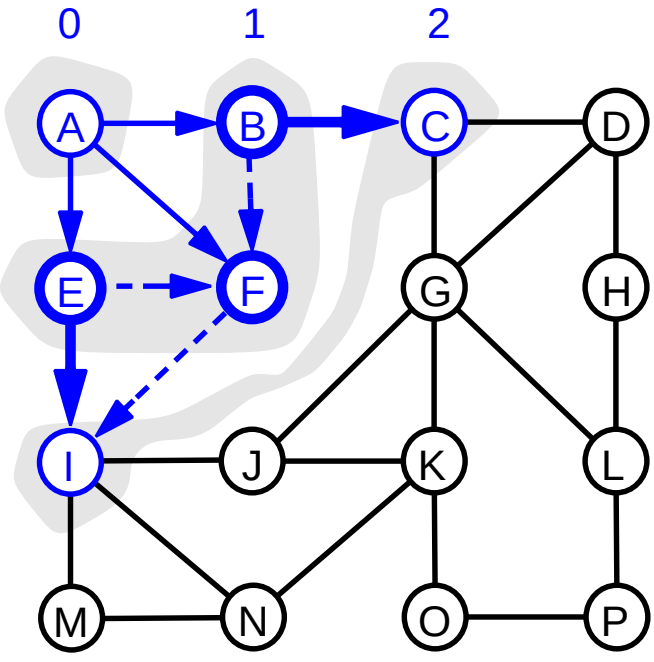
a)



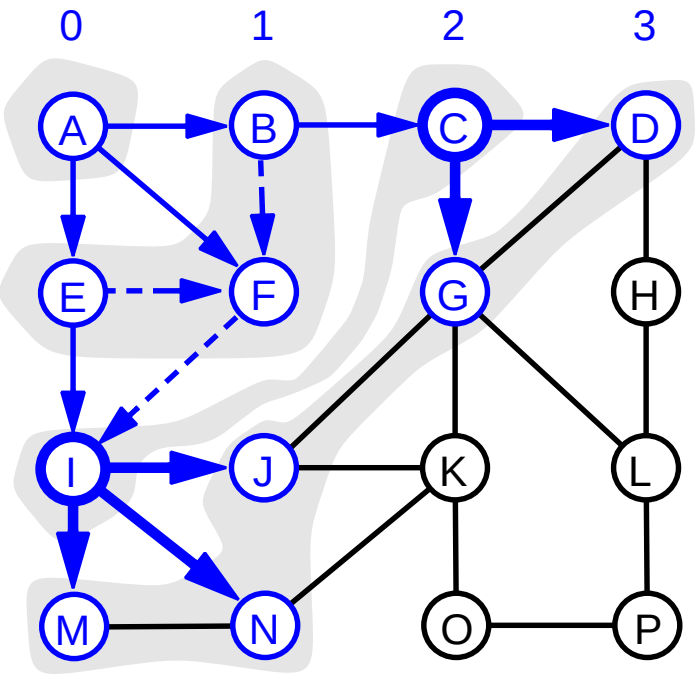
b)



c)

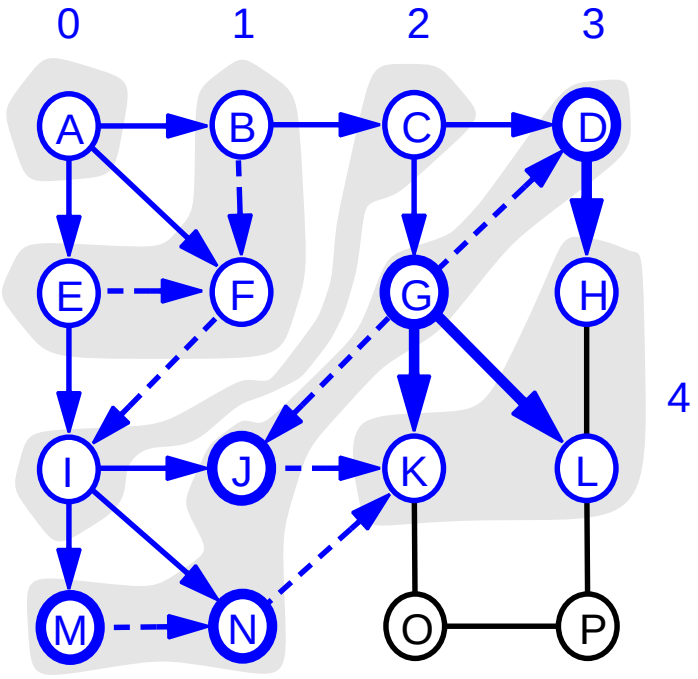


d)

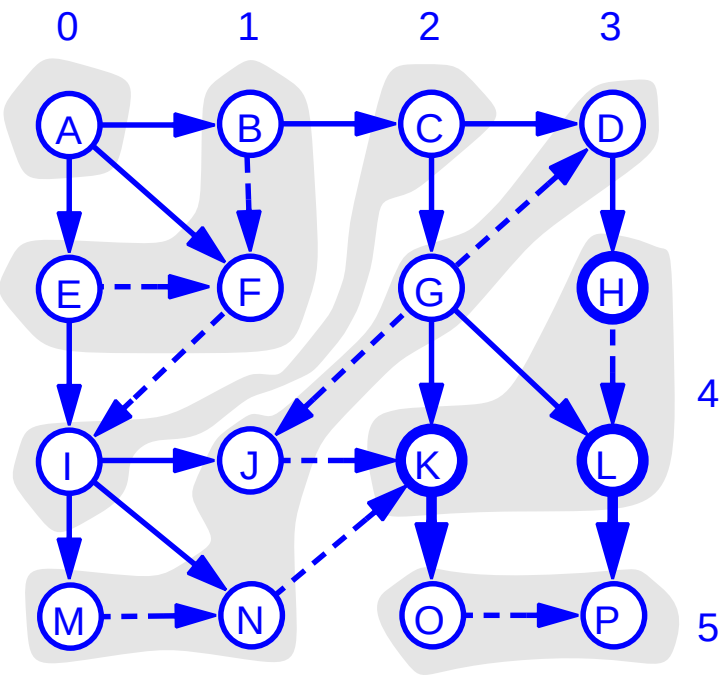


More BFS

e)



f)



BFS Pseudo-Code

```
Algorithm BFS(s):
  Input: A vertex s in a graph
  Output: A labeling of the edges as "discovery" edges
    and "cross edges"
  initialize container L_0 to contain vertex s
  i ← 0
  while L_i is not empty do
    create container L_{i+1} to initially be empty
    for each vertex v in L_i do
      for each edge e incident on v do
        if edge e is unexplored then
          let w be the other endpoint of e
          if vertex w is unexplored then
            label e as a discovery edge
            insert w into L_{i+1}
          else
            label e as a cross edge
    i ← i + 1
```

Properties of BFS

- Proposition:Let G be an undirected graph on which a BFS traversal starting at vertex s has been performed. Then

  -The traversal visits all vertices in the connected component of s.

  -The discovery-edges form a spanning tree T, which we call the BFS tree, of the connected component of s

  -For each vertex v at level i,the path of the BFS tree T between s and v has i edges, and any other path of G between s and v has at least i edges.

  - I f(u, v) is an edge that is not in the BFS tree, then the level numbers of u and v differ by at most one.

- Proposition: Let G be a graph with n vertices and m edges. A BFS traversal of G takes time $O(n + m)$. Also, there exist $O(n + m)$ time algorithms based on BFS for the following problems:

  -Testing whether G is connected.

  -Computing a spanning tree of G

  -Computing the connected components of G

  -Computing, for every vertex v of G, the minimum number of edges of any path between s and v.