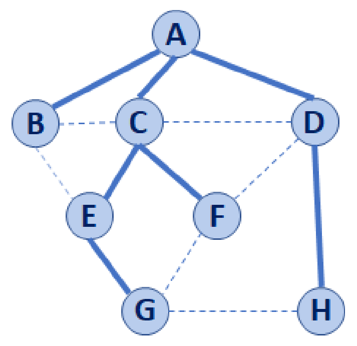
CS 225

#37: Graph Traversal - DFS

November 29, 2017 · *Wade Fagen-Ulmschneider*

BFS Graph Traversal



d	p	v	Adjacent
0	A	A	C B D
1	A	B	A C E
1	A	C	B A D E F
1	A	D	A C F H
2	C	E	B C G
2	C	F	C D G
3	E	G	E F H
2	D	H	D G

Pseudocode for BFS

```
BFS(G):
  Input: Graph, G
  Output: A labeling of the edges on
      G as discovery and cross edges

  foreach (Vertex v : G.vertices()):
    setLabel(v, UNEXPLORED)
  foreach (Edge e : G.edges()):
    setLabel(e, UNEXPLORED)
  foreach (Vertex v : G.vertices()):
    if getLabel(v) == UNEXPLORED:
       BFS(G, v)

BFS(G, v):
  Queue q
  setLabel(v, VISITED)
  q.enqueue(v)

  while !q.empty():
    v = q.dequeue()
    foreach (Vertex w : G.adjacent(v)):
      if getLabel(w) == UNEXPLORED:
         setLabel(v, w, DISCOVERY)
         setLabel(w, VISITED)
         q.enqueue(w)
      elseif getLabel(v, w) == UNEXPLORED:
         setLabel(v, w, CROSS)
```

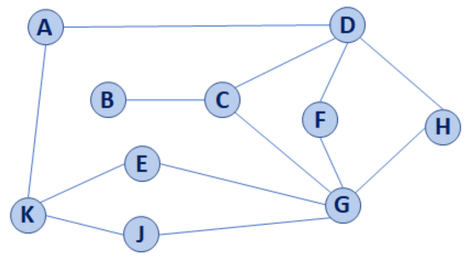
BST Graph Observations

1. Does our implementation handle disjoint graphs? How?
   a. How can we modify our code to count components?
2. Can our implementation detect a cycle? How?
   a. How can we modify our code to store update a private member variable `cycleDetected_`?
3. What is the running time of our algorithm?
4. What is the shortest path between **A** and **H**?
5. What is the shortest path between **E** and **H**?
   a. What does that tell us about BFS?
6. What does a cross edge tell us about its endpoints?
7. What structure is made from discovery edges in **G**?

Big Ideas: Utility of a BFS Traversal

Obs. 1: Traversals can be used to count components.
Obs. 2: Traversals can be used to detect cycles.
Obs. 3: In BFS, **d** provides the shortest distance to every vertex.
Obs. 4: In BFS, the endpoints of a cross edge never differ in distance, d, by more than 1: **$|d(u) - d(v)| = 1$**

Depth First Search – A Modification to BFS



Two types of edges: 1.

2.

Running Time of DFS:

Labeling:

- Vertex:
- Edge:

Queries:

- Vertex:
- Edge:

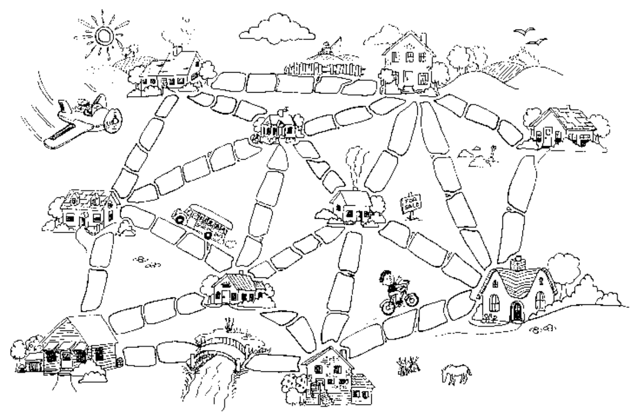
Pseudocode for DFS

```
BFS(G):
  Input: Graph, G
  Output: A labeling of the edges on
      G as discovery and cross edges

  foreach (Vertex v : G.vertices()):
    setLabel(v, UNEXPLORED)
  foreach (Edge e : G.edges()):
    setLabel(e, UNEXPLORED)
  foreach (Vertex v : G.vertices()):
    if getLabel(v) == UNEXPLORED:
       BFS(G, v)

BFS(G, v):
  Queue q
  setLabel(v, VISITED)
  q.enqueue(v)

  while !q.empty():
    v = q.dequeue()
    foreach (Vertex w : G.adjacent(v)):
      if getLabel(w) == UNEXPLORED:
         setLabel(v, w, DISCOVERY)
         setLabel(w, VISITED)
         q.enqueue(w)
      elseif getLabel(v, w) == UNEXPLORED:
         setLabel(v, w, CROSS)
```



"The Muddy City" by CS Unplugged, **Creative Commons BY-NC-SA 4.0**

CS 225 – Things To Be Doing:

1. Exam #11 (theory) is ongoing
2. MP7 released (+14 EC due on Monday!)
3. lab_dictionary due Wednesday at 7:00pm
4. lab_graphs starts Wednesday
5. Multi-day "puzzle" POTDs available M/W/F