Graph ADT

- Initialize(n): Initialize a graph with n vertices
- AddEdge(v, w): Adds an edge between v and w
- RemoveEdge(v, w): If exists, removes the edge between v and w
- IsAdjacent?(v, w): Returns true if v and w are adjacent
- GetNeighbors(v): Returns the set of all adjacent vertices of v

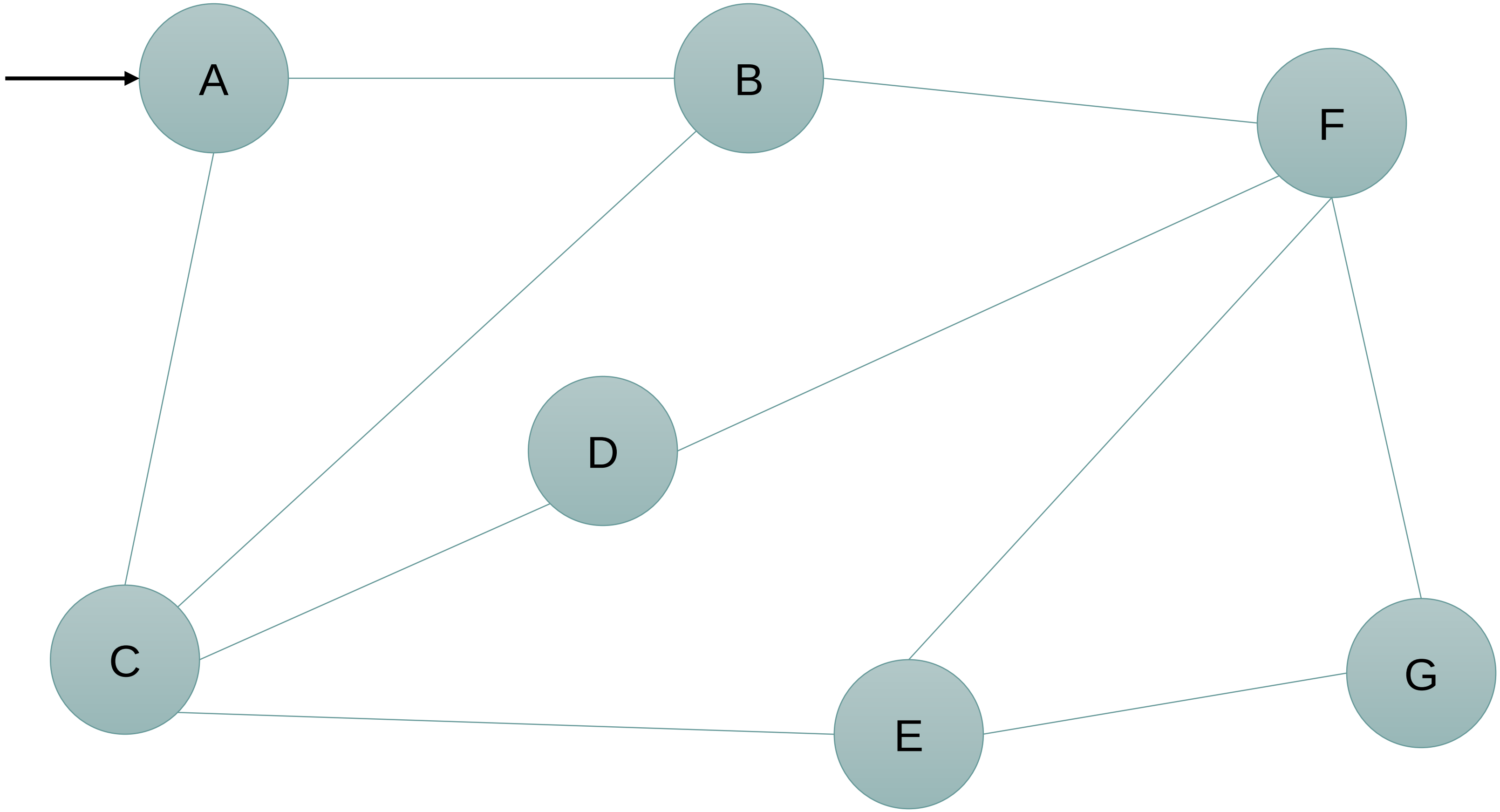
Graph Algorithms

- Breadth-first search (BFS)
- Depth-first search (DFS)
- Detect cycles

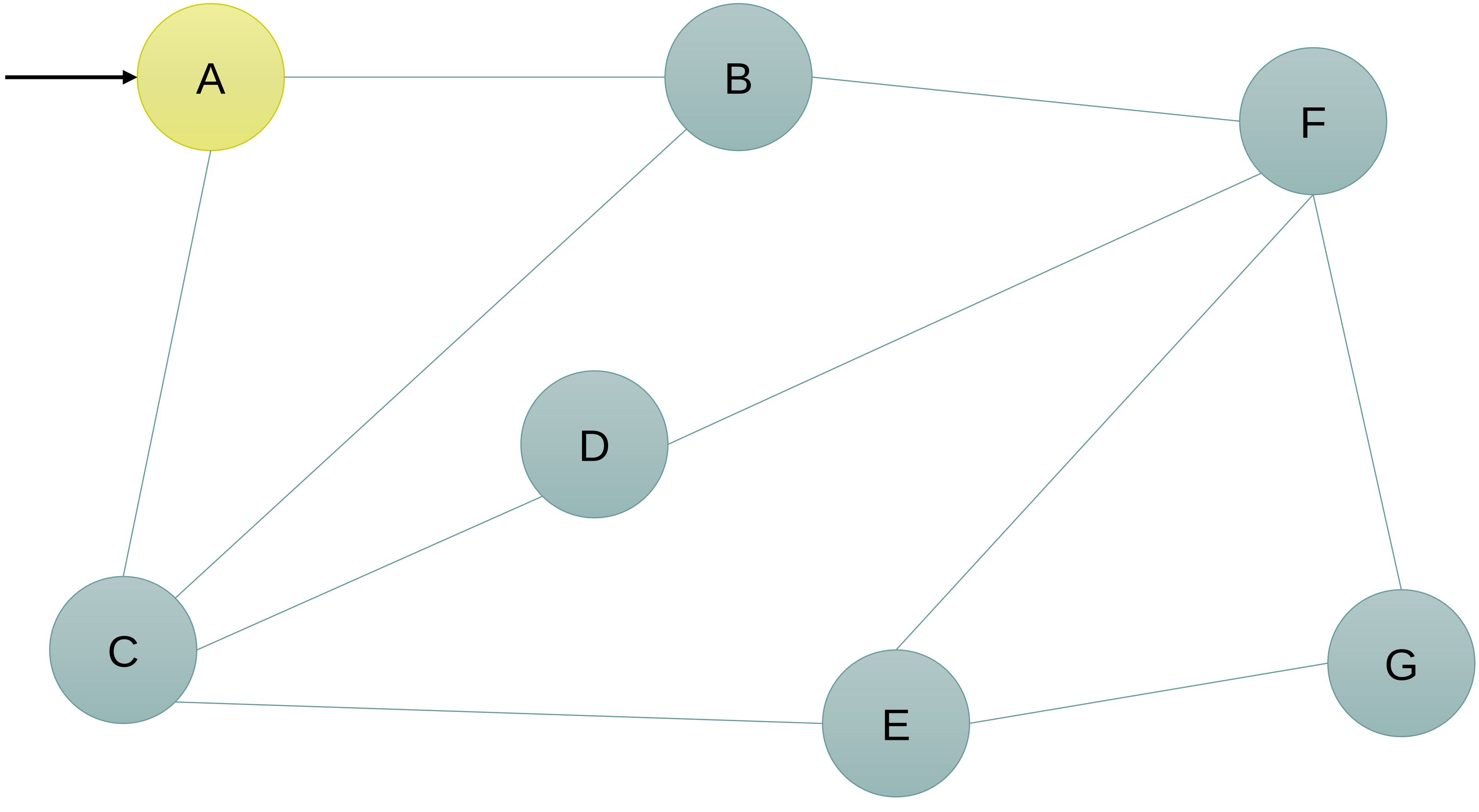
Breadth-first Search (BFS)

- An algorithm to visit all the vertices reachable for one starting vertex
- Visit the starting vertex (v)
- Visit the neighbors of (v)
- Visit the second-degree neighbors of (v)
- …
- Until no more vertices to visit

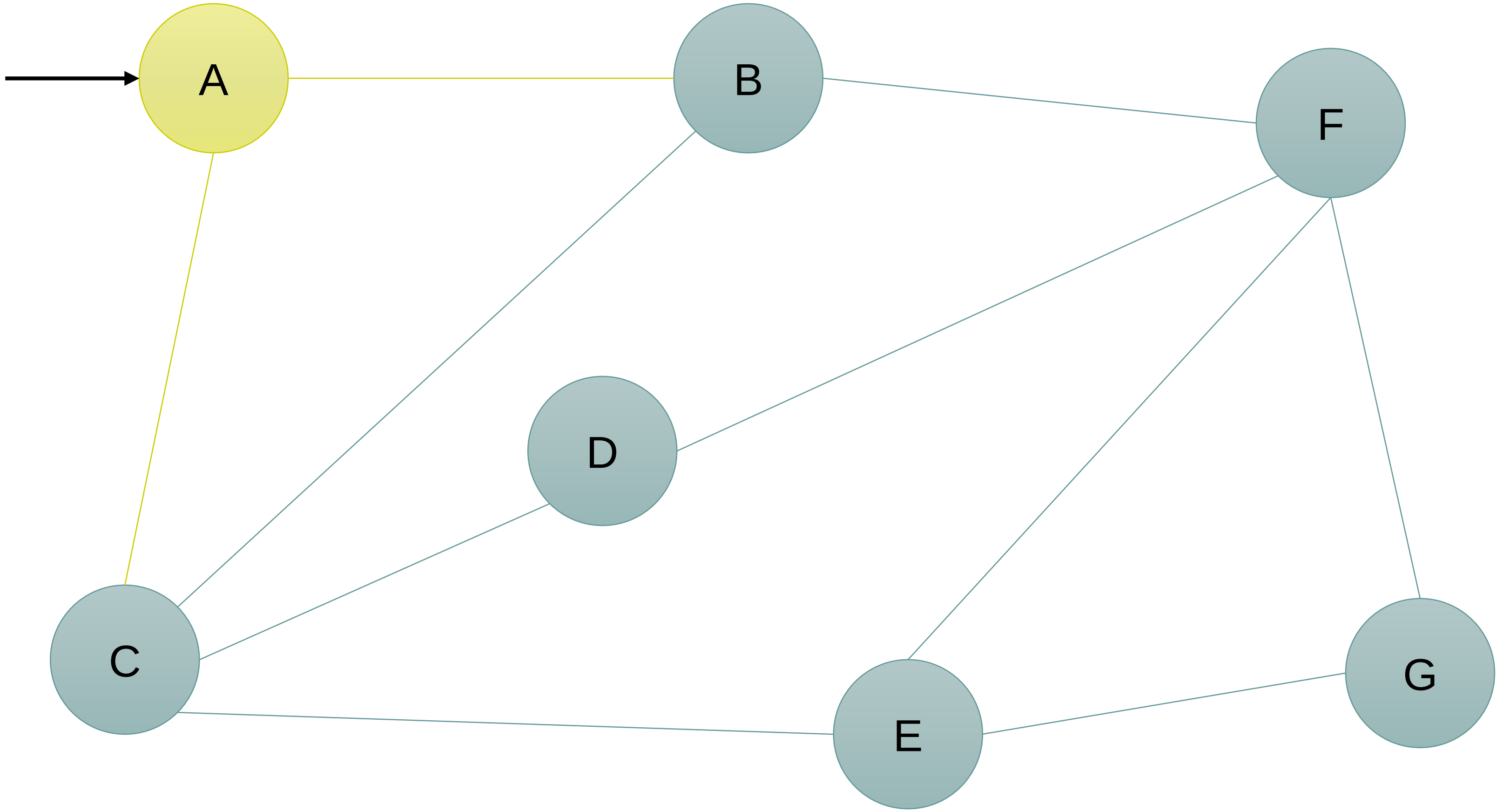
Breadth-first Search (BFS)



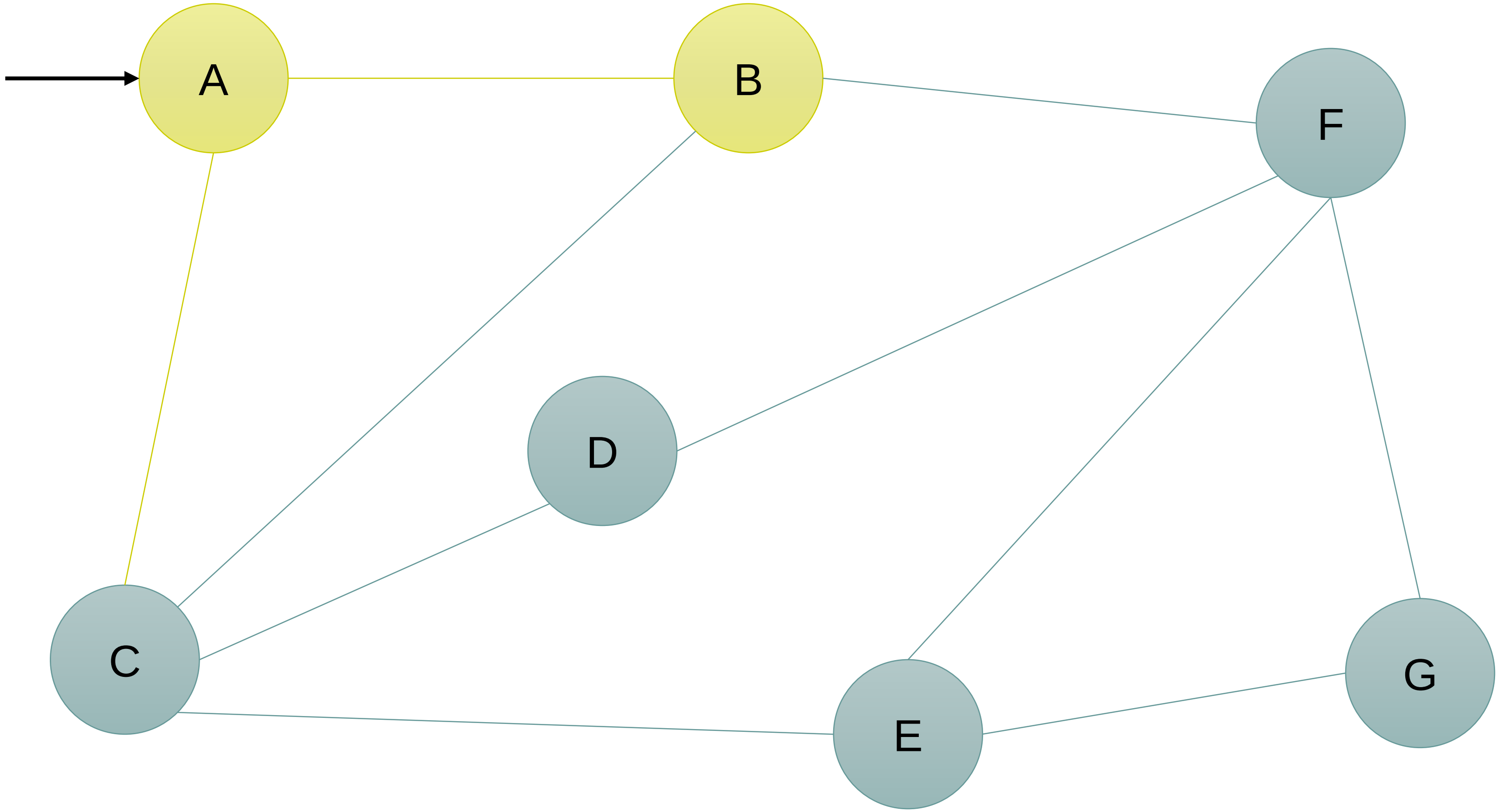
Breadth-first Search (BFS)



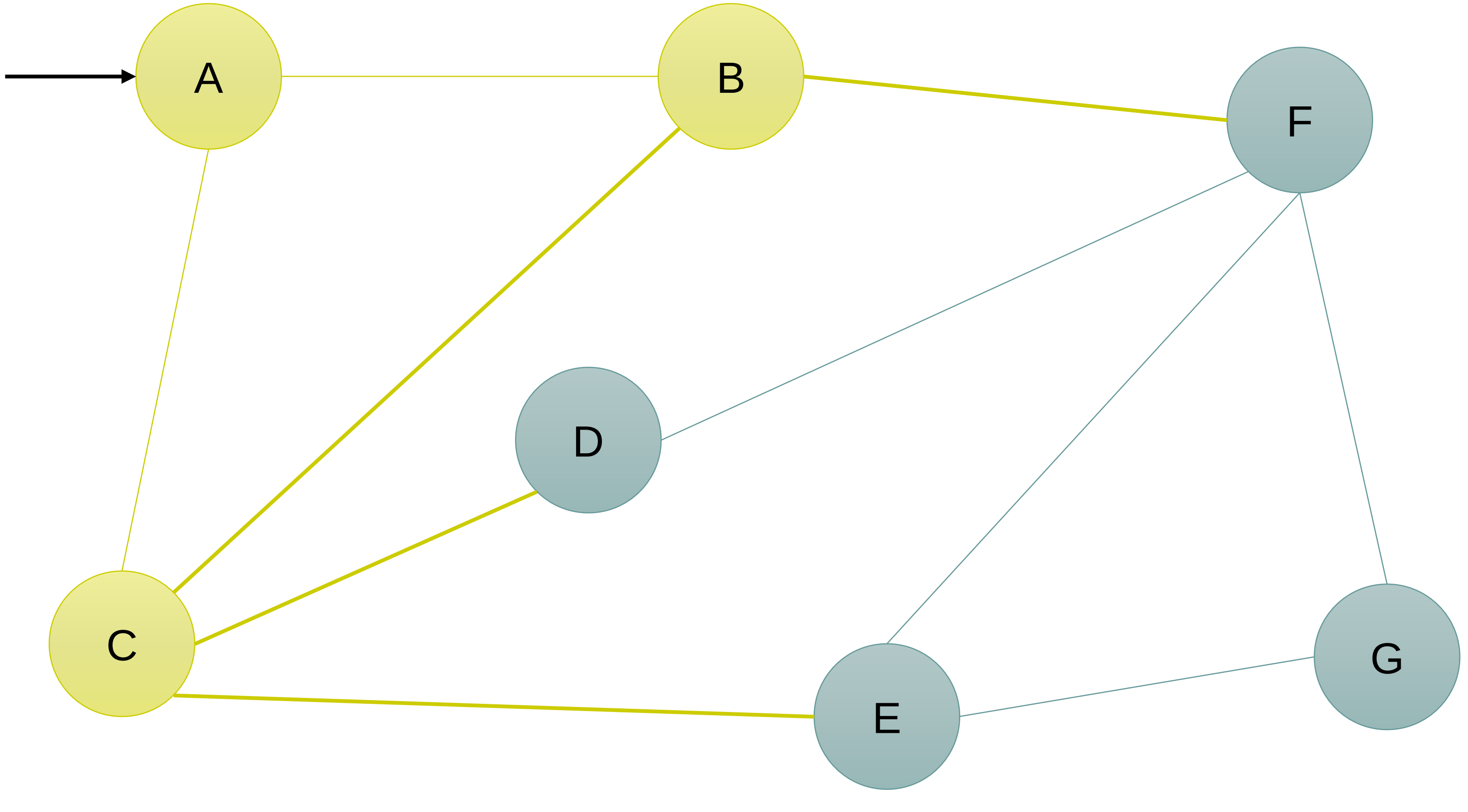
Breadth-first Search (BFS)



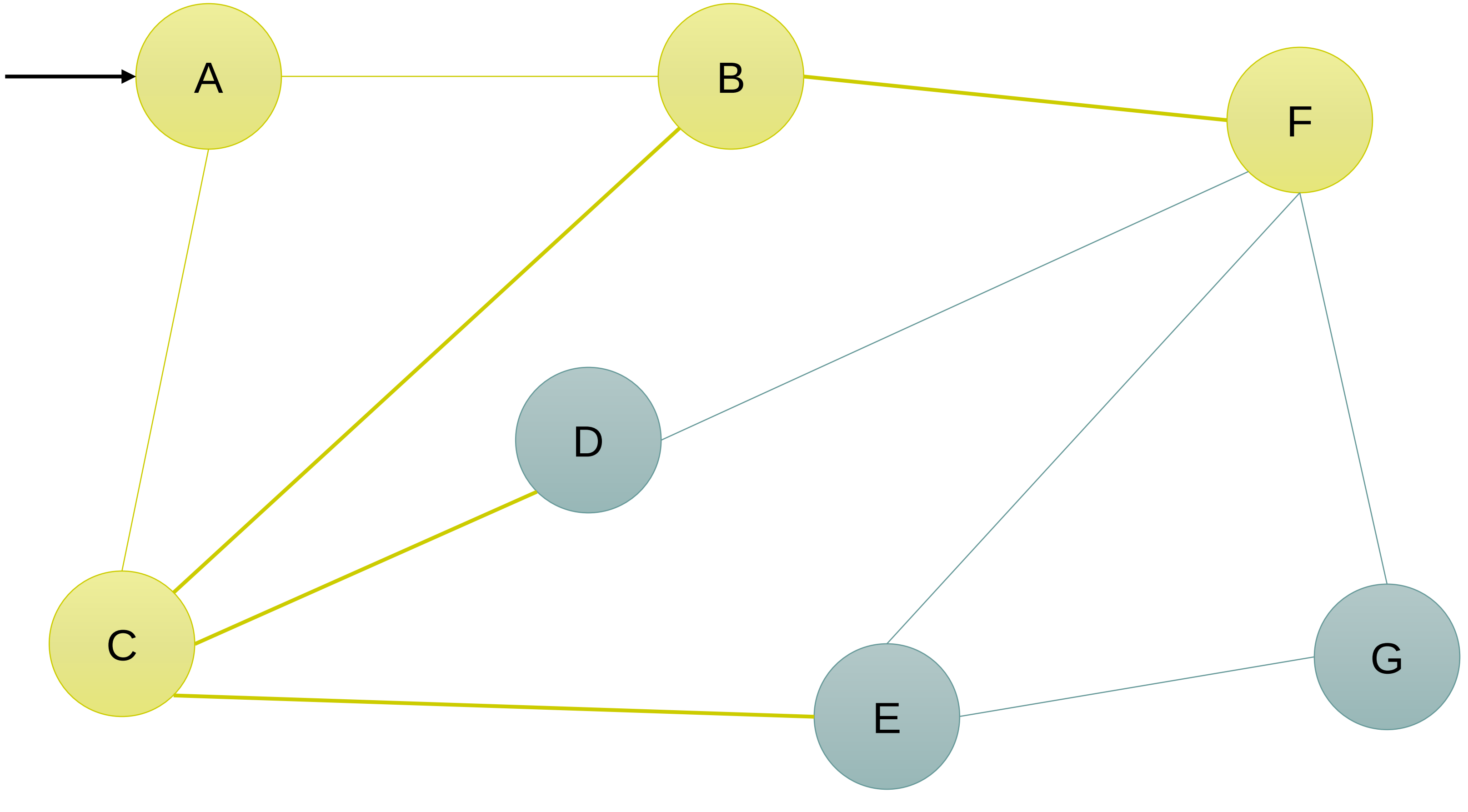
Breadth-first Search (BFS)



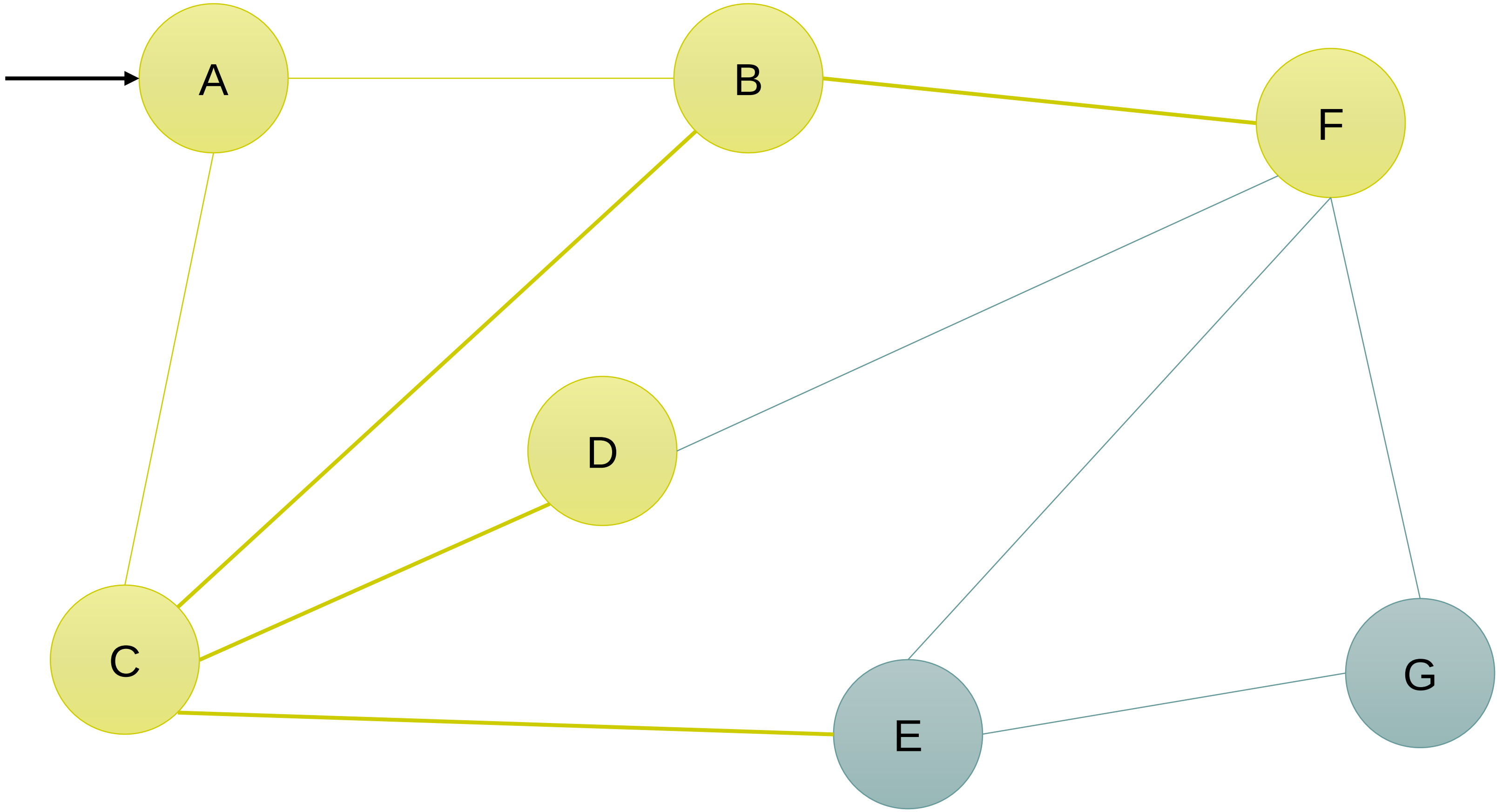
Breadth-first Search (BFS)



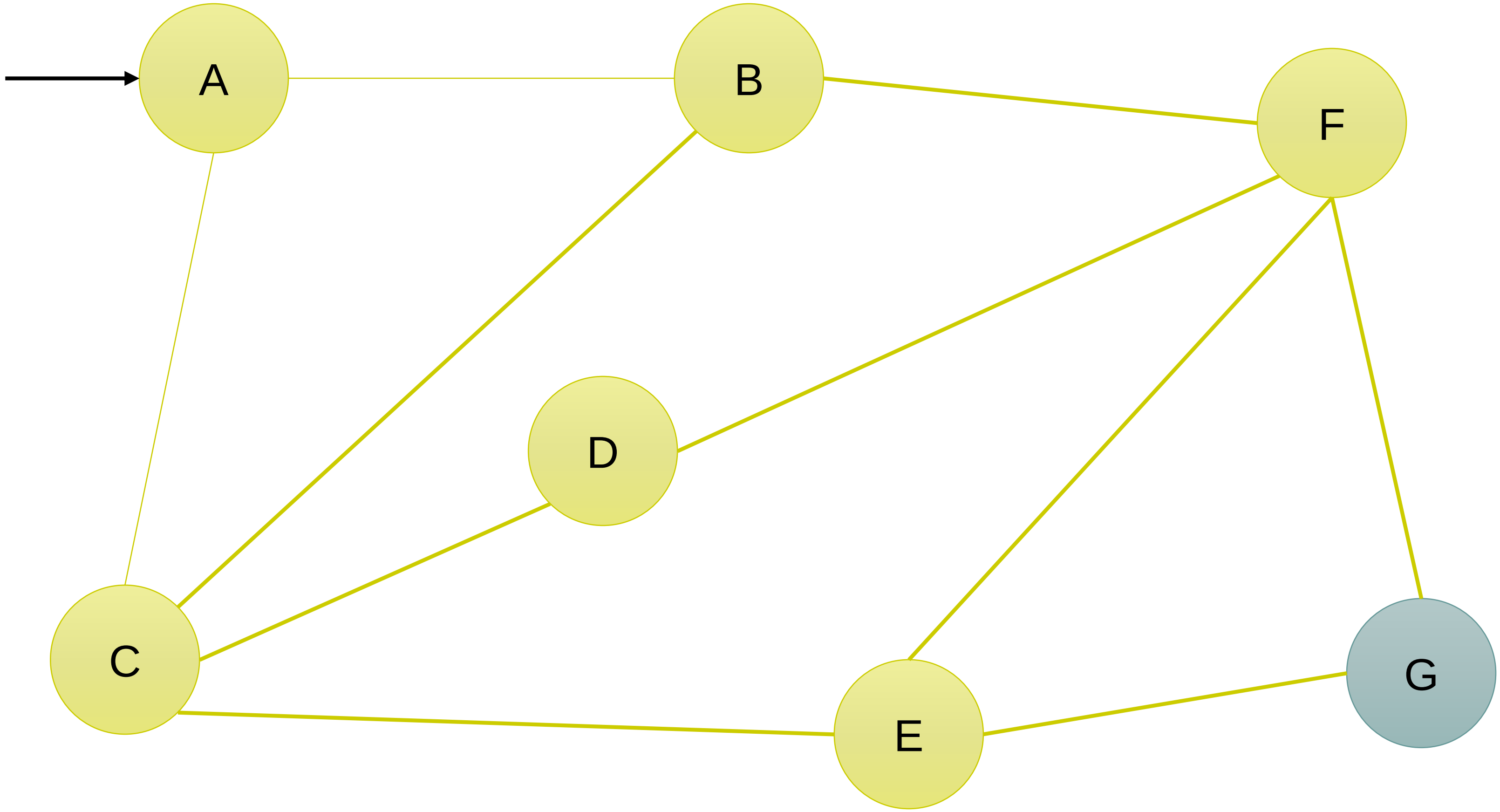
Breadth-first Search (BFS)



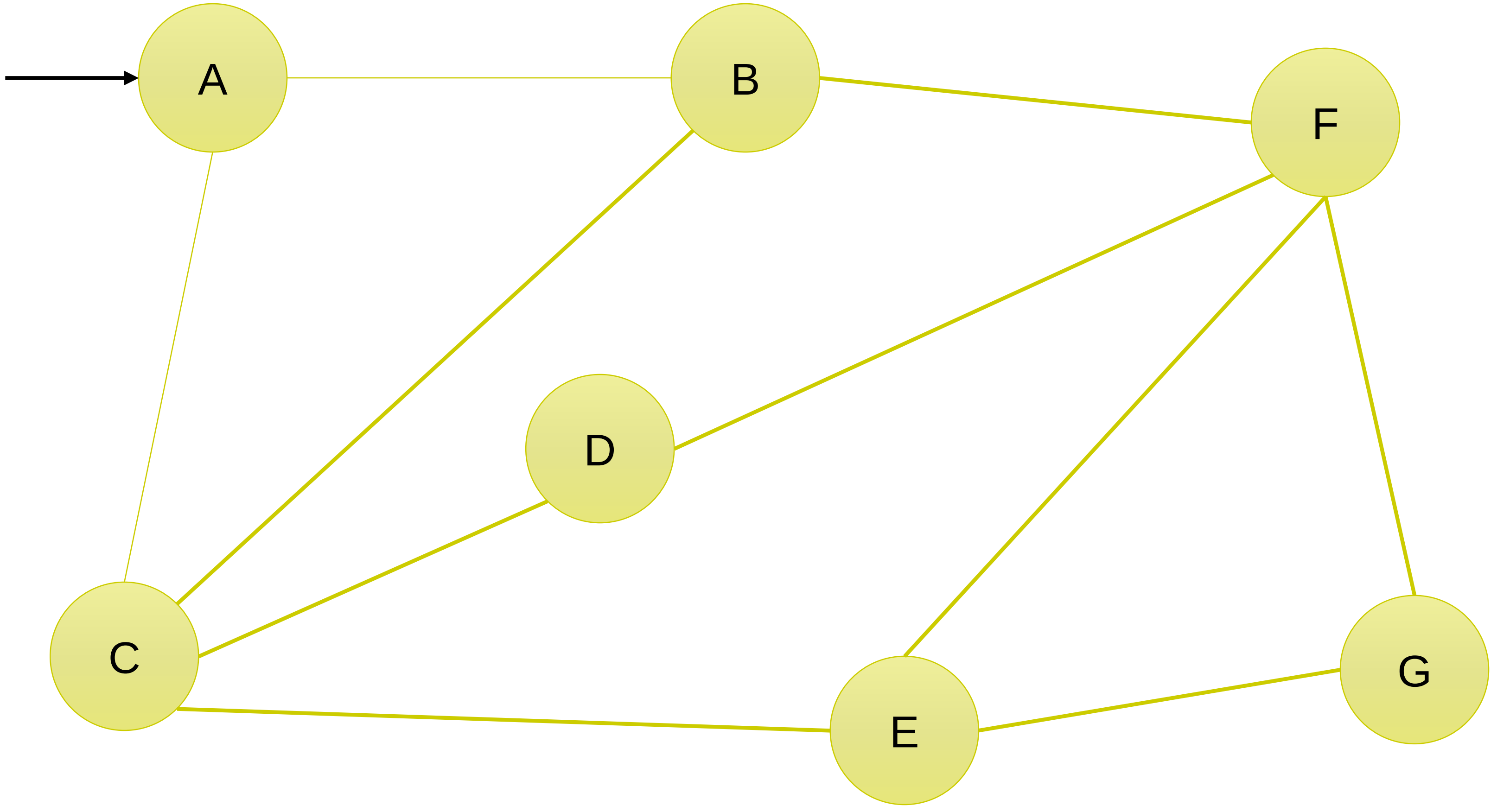
Breadth-first Search (BFS)



Breadth-first Search (BFS)



Breadth-first Search (BFS)



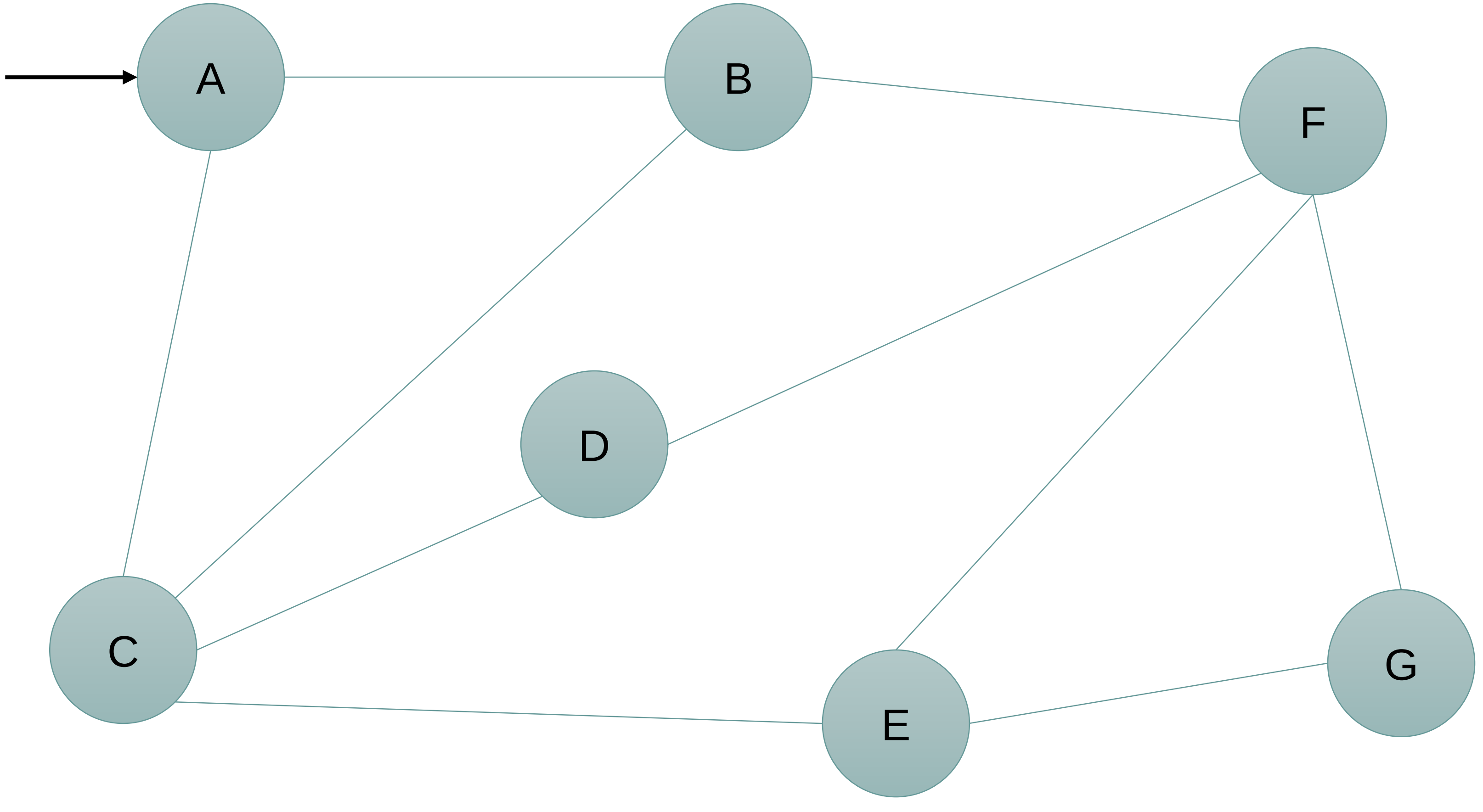
Breadth-first Search (BFS)

- In some cases, we would like to keep track of the path length from the starting vertex to each visited vertex
- The visited vertices and edges can be used to create a BFS-tree representation of the graph.

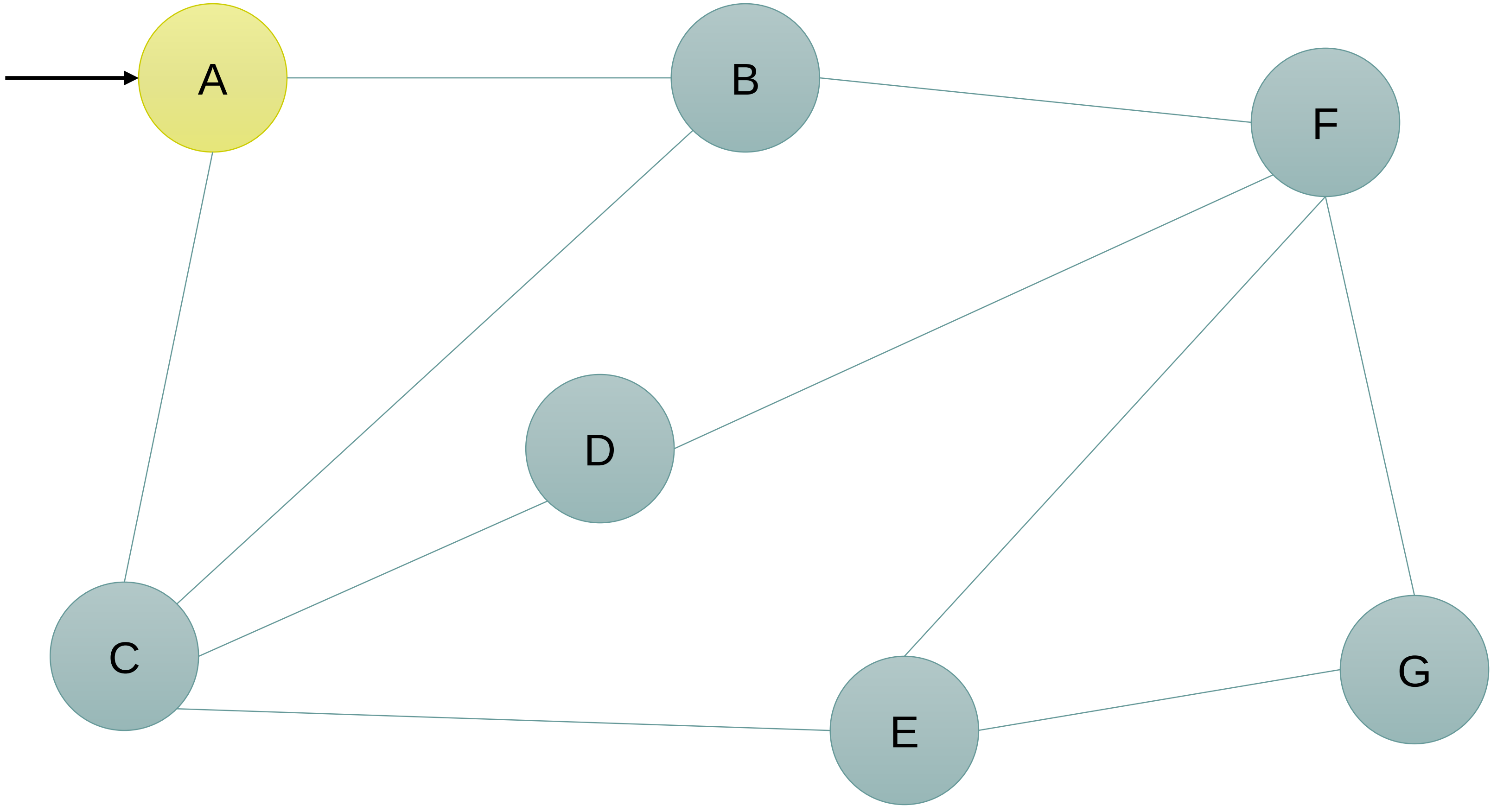
Depth-first Search (DFS)

- An algorithm to visit all the vertices reachable for one starting vertex
- Visit the starting vertex (v)
- Visit one neighbor of v
- Visit as much as possible from that neighbor until moving to another neighbor
- …
- Until all vertices reachable from v are visited

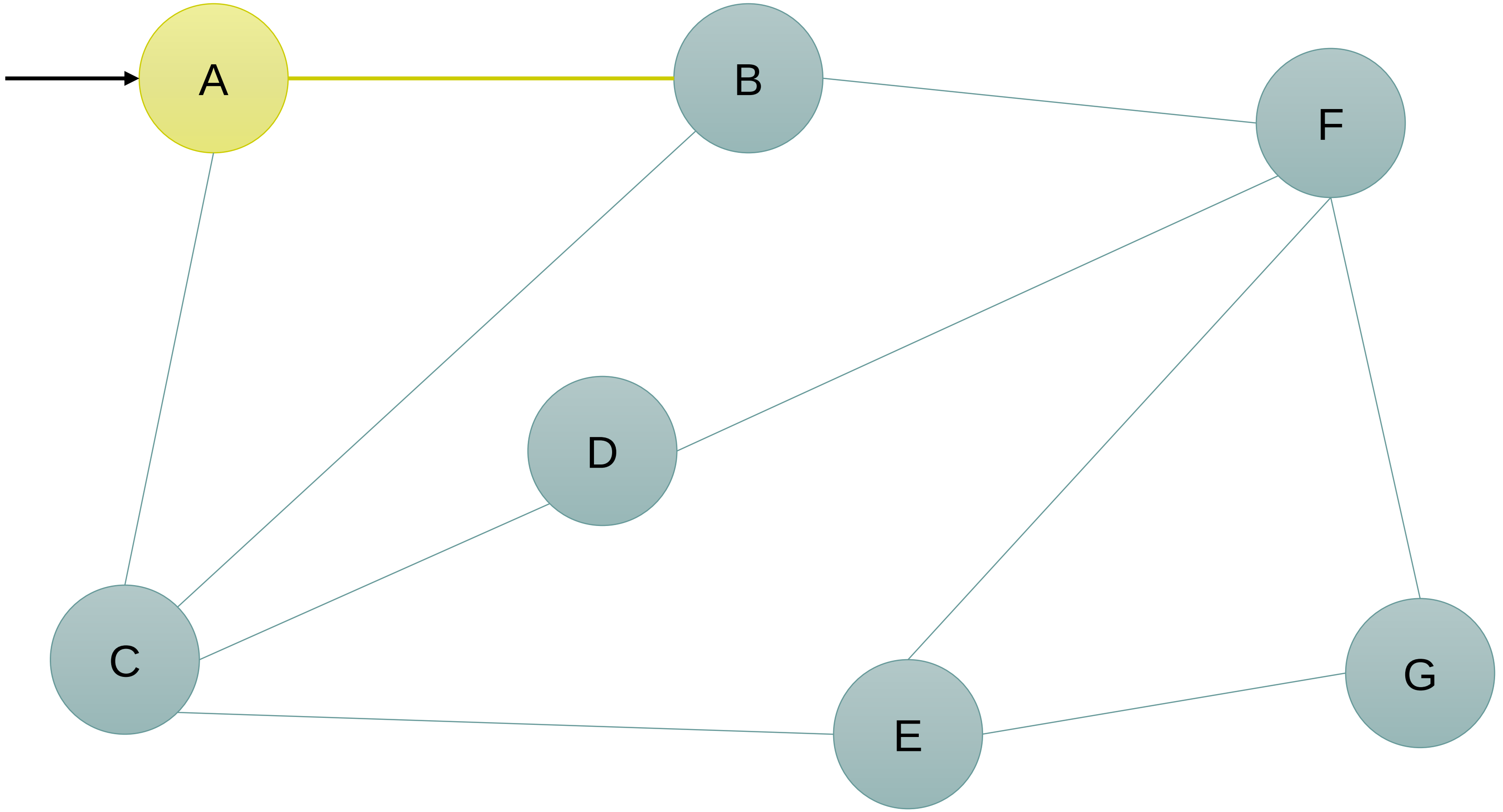
Depth-first Search (DFS)



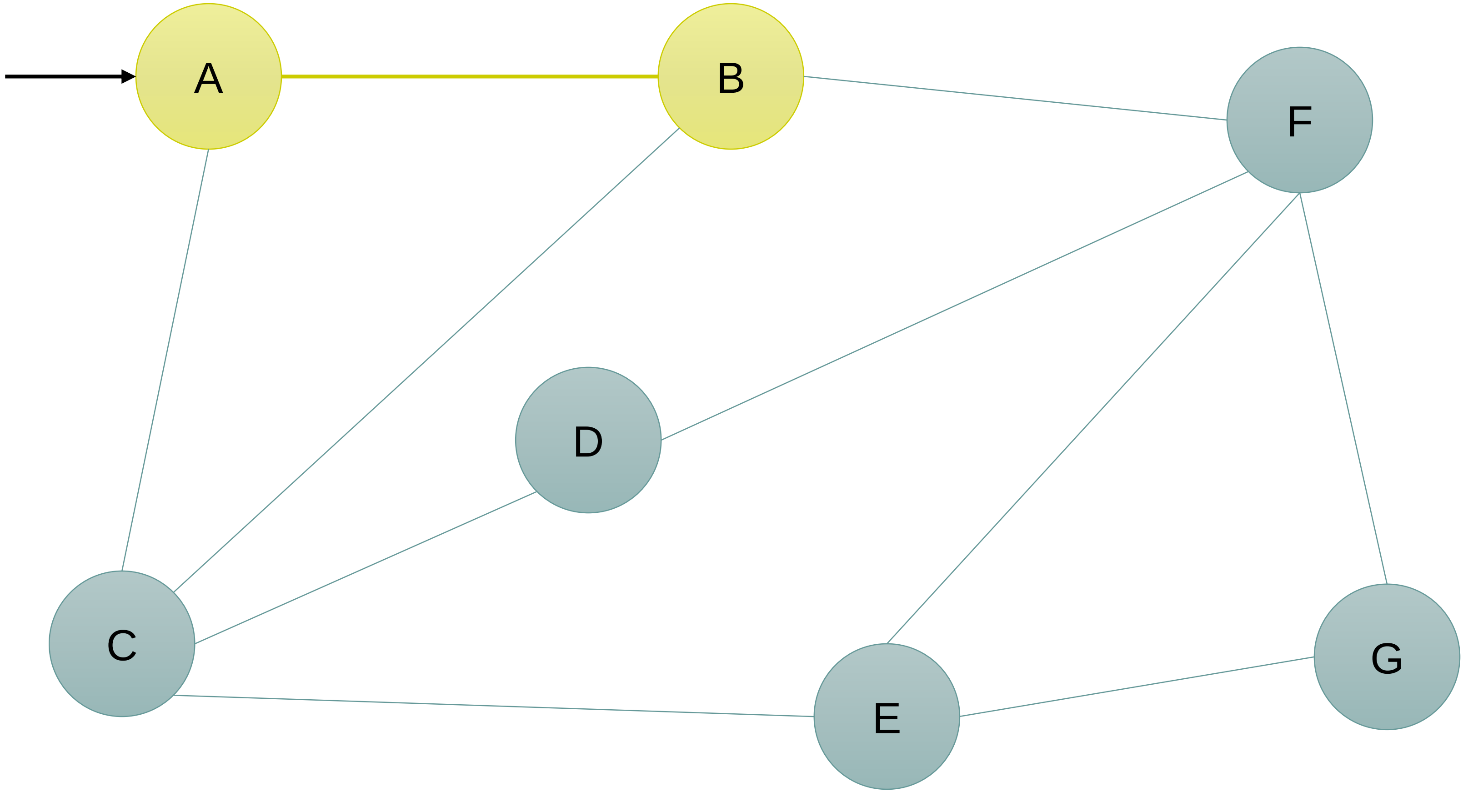
Depth-first Search (DFS)



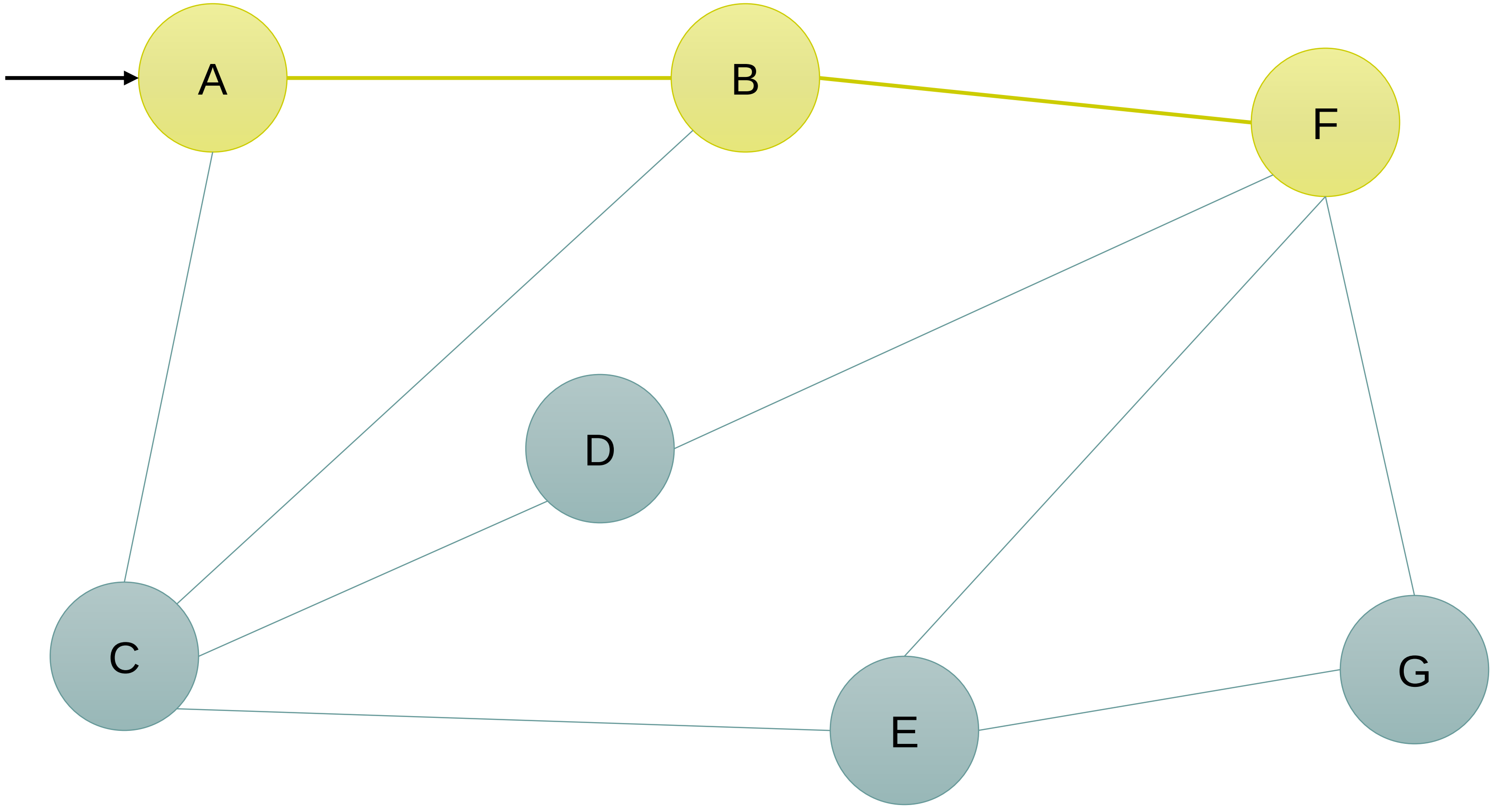
Depth-first Search (DFS)



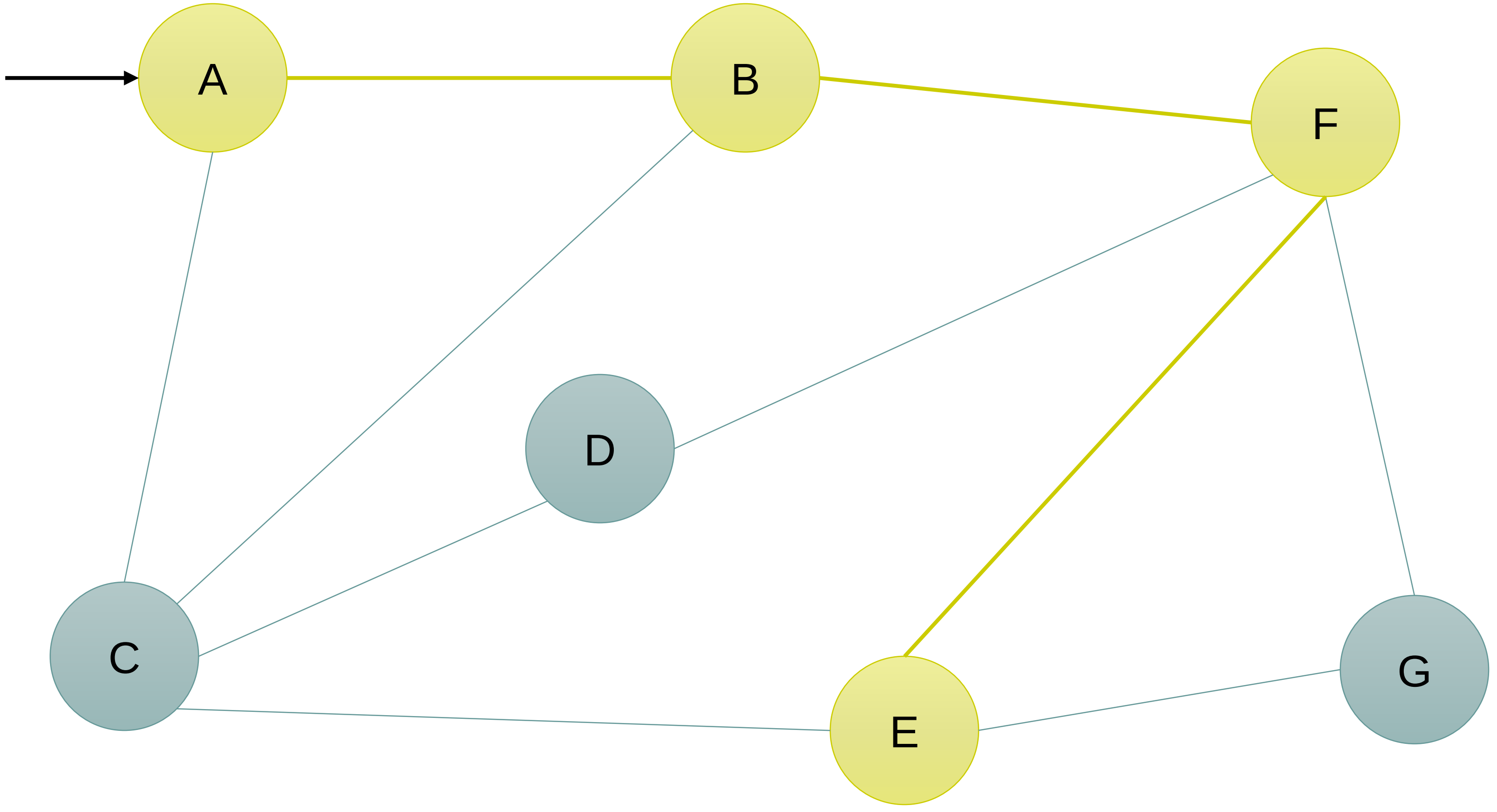
Depth-first Search (DFS)



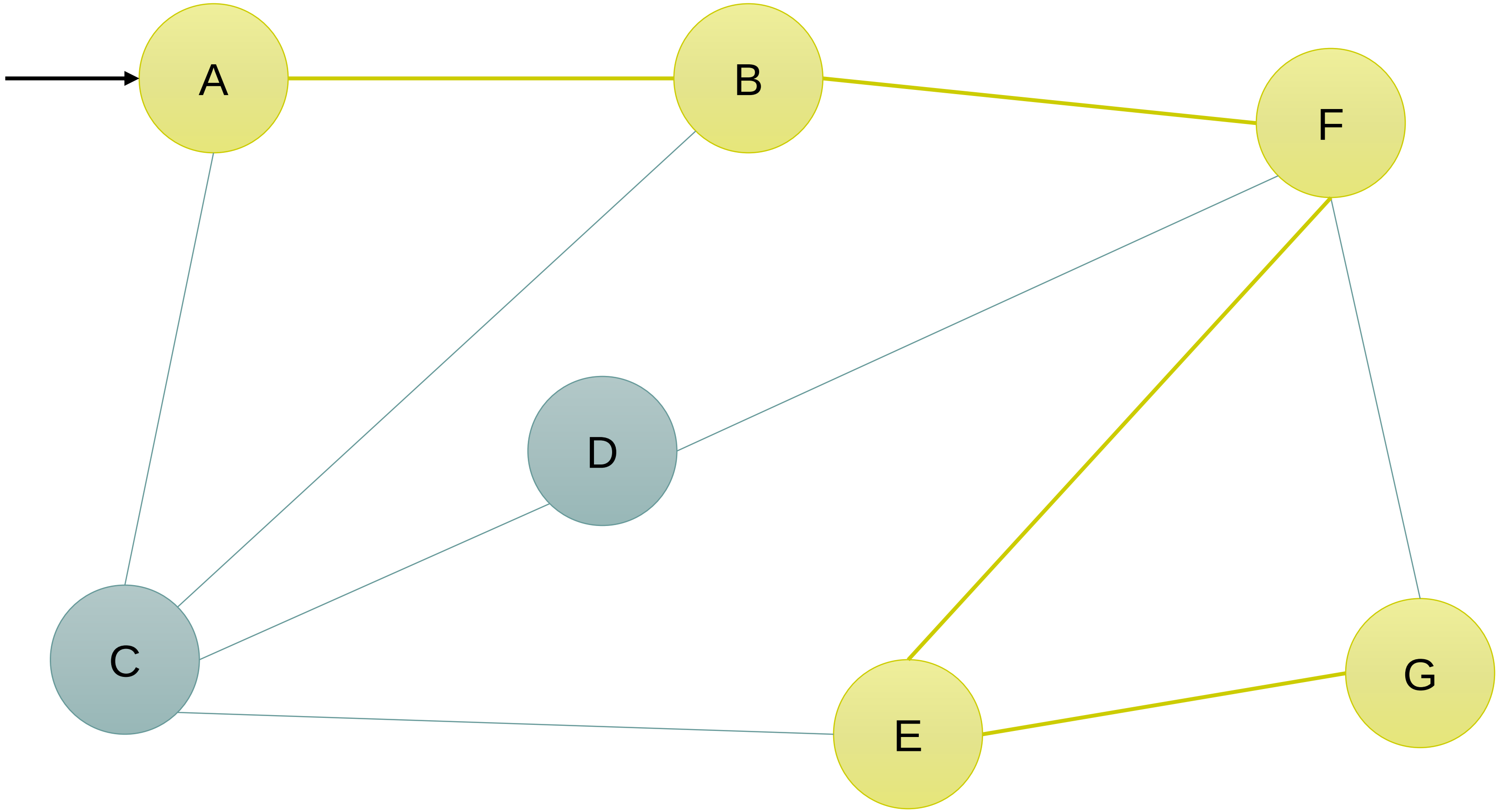
Depth-first Search (DFS)



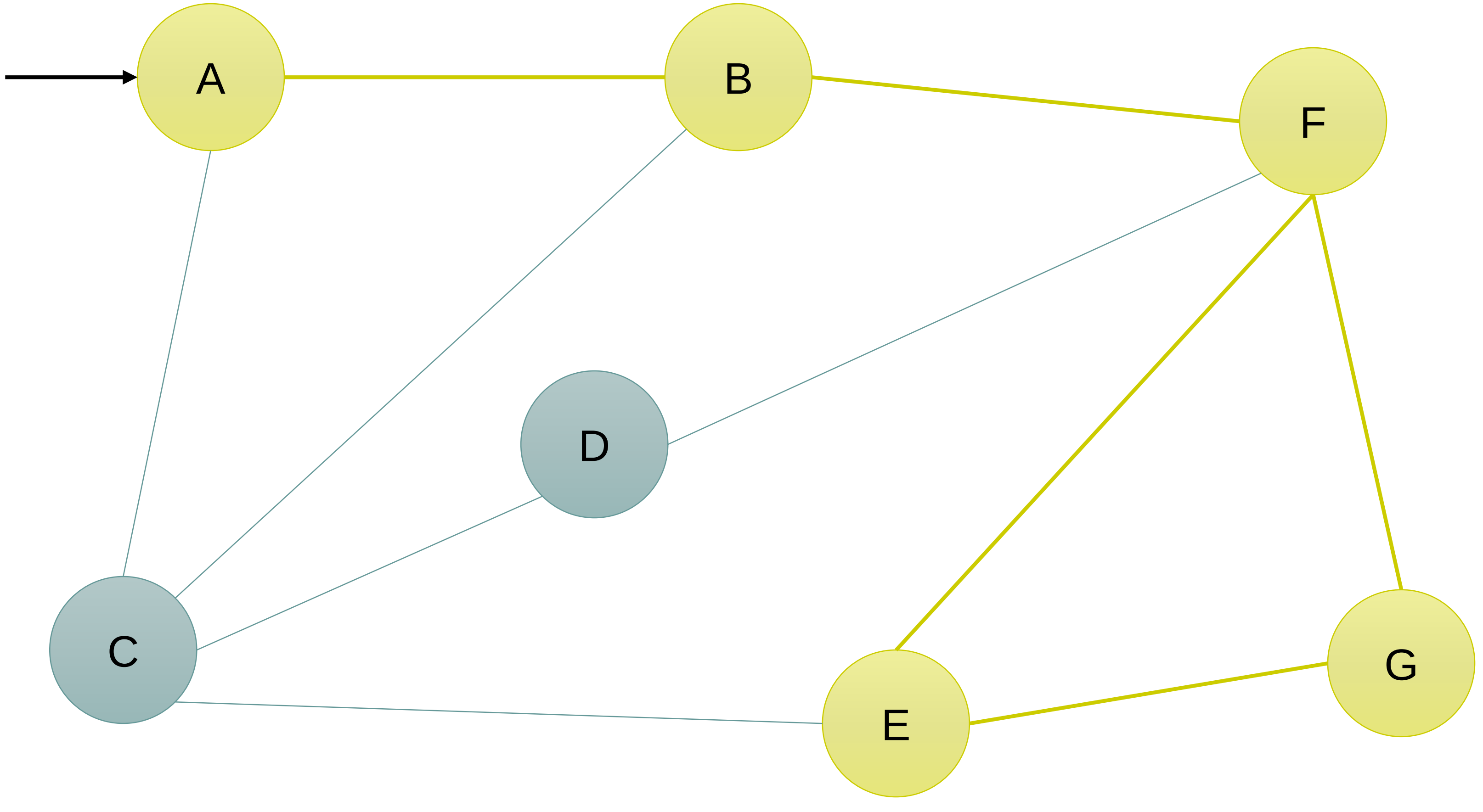
Depth-first Search (DFS)



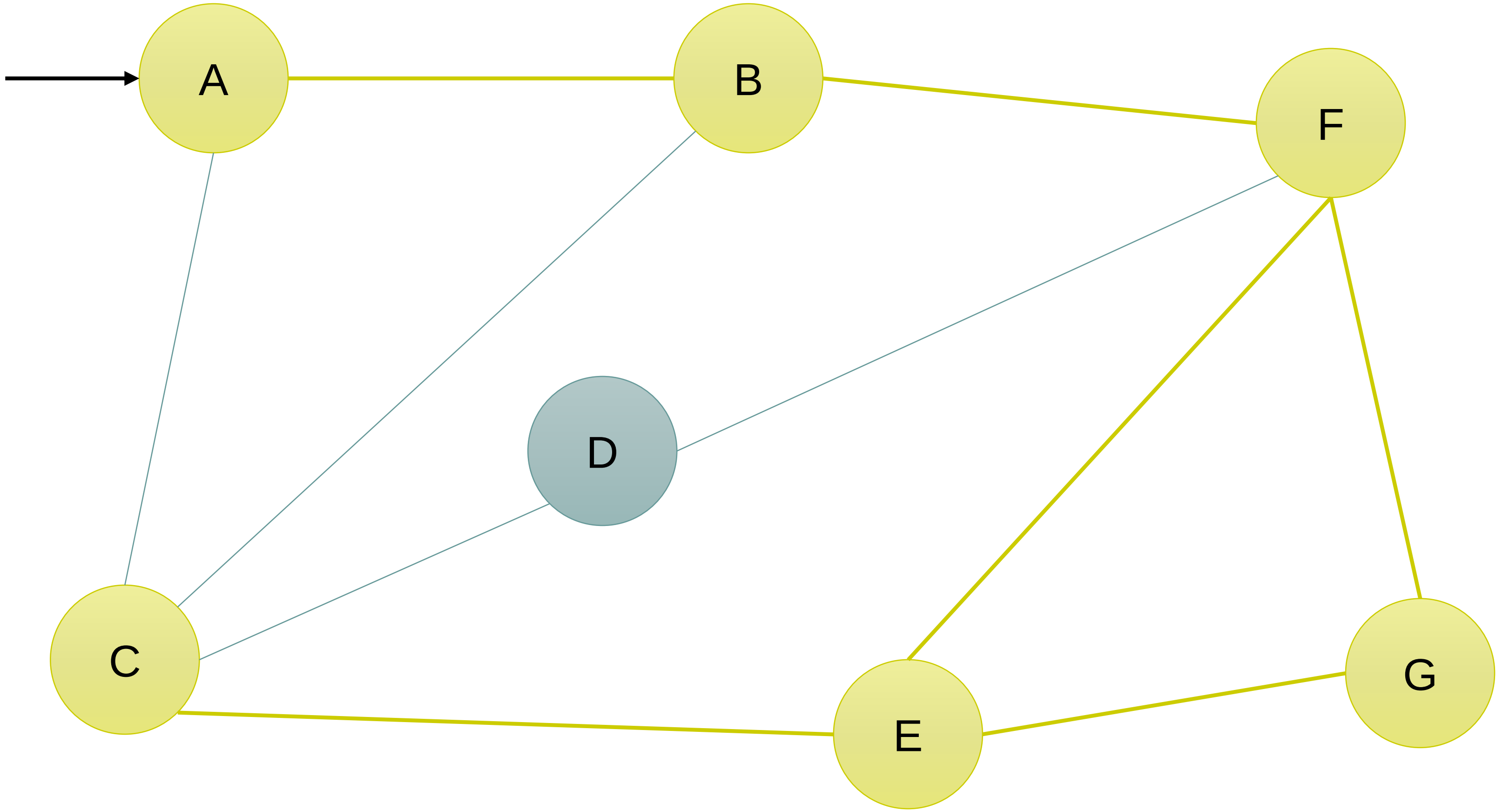
Depth-first Search (DFS)



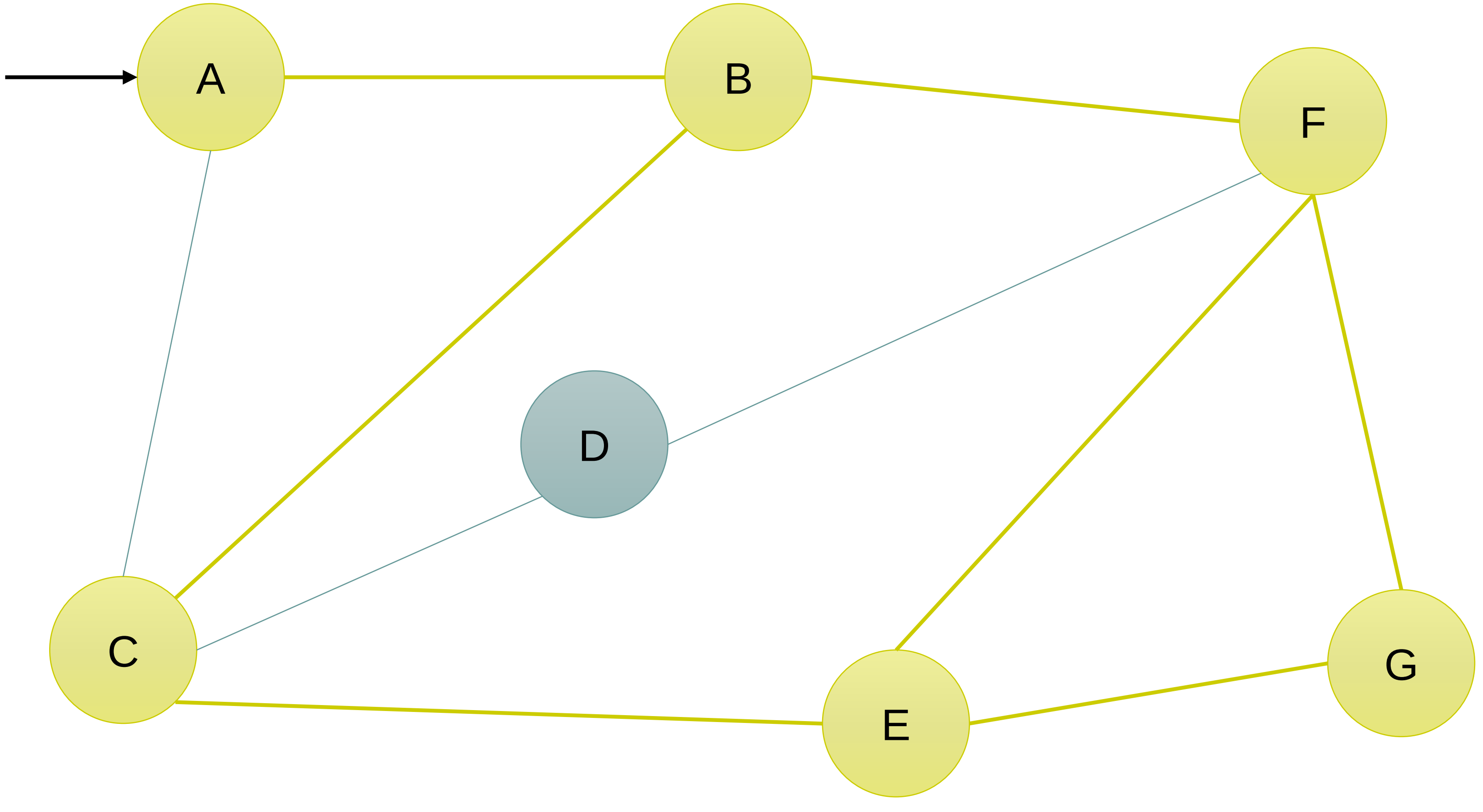
Depth-first Search (DFS)



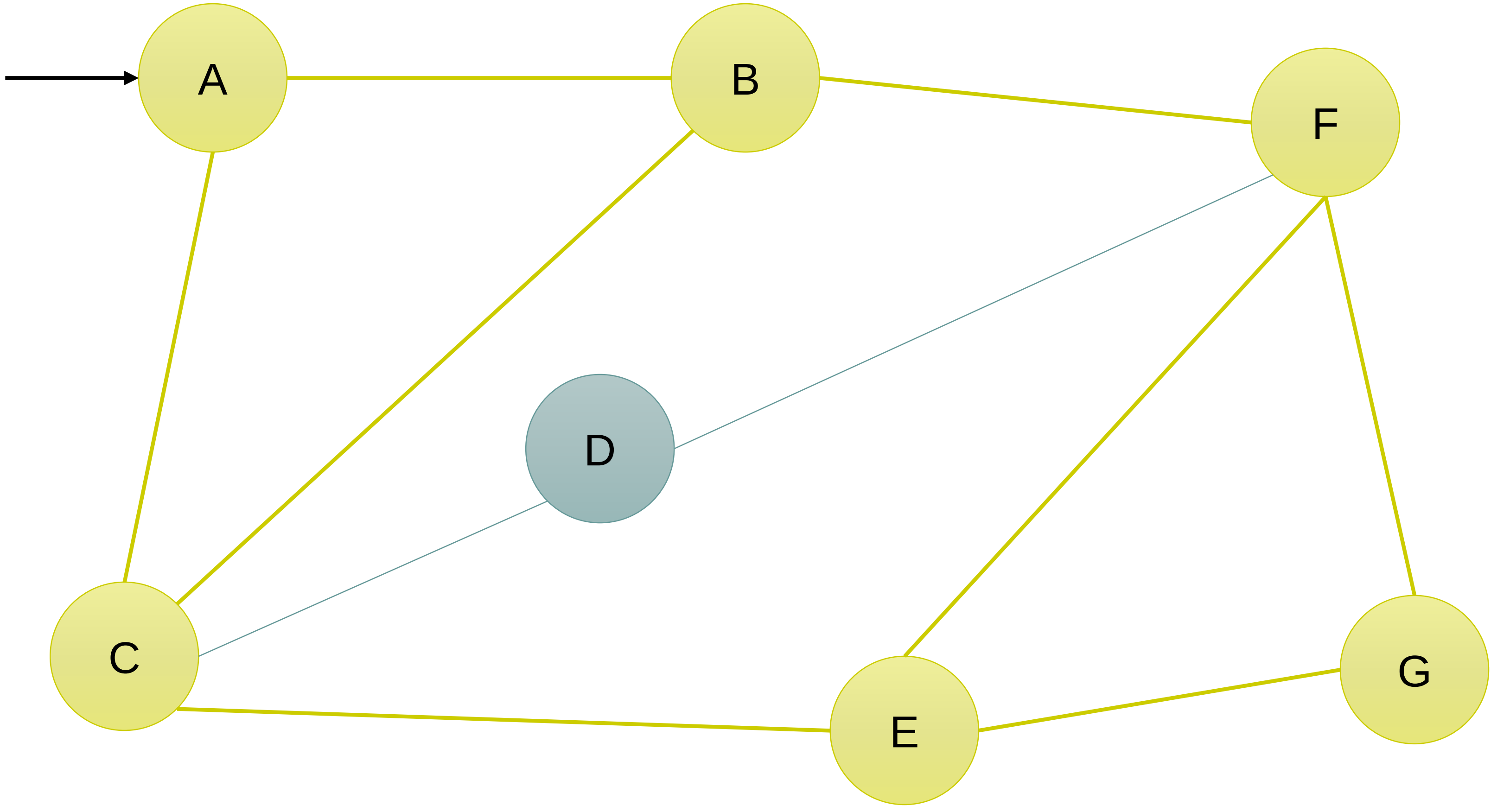
Depth-first Search (DFS)



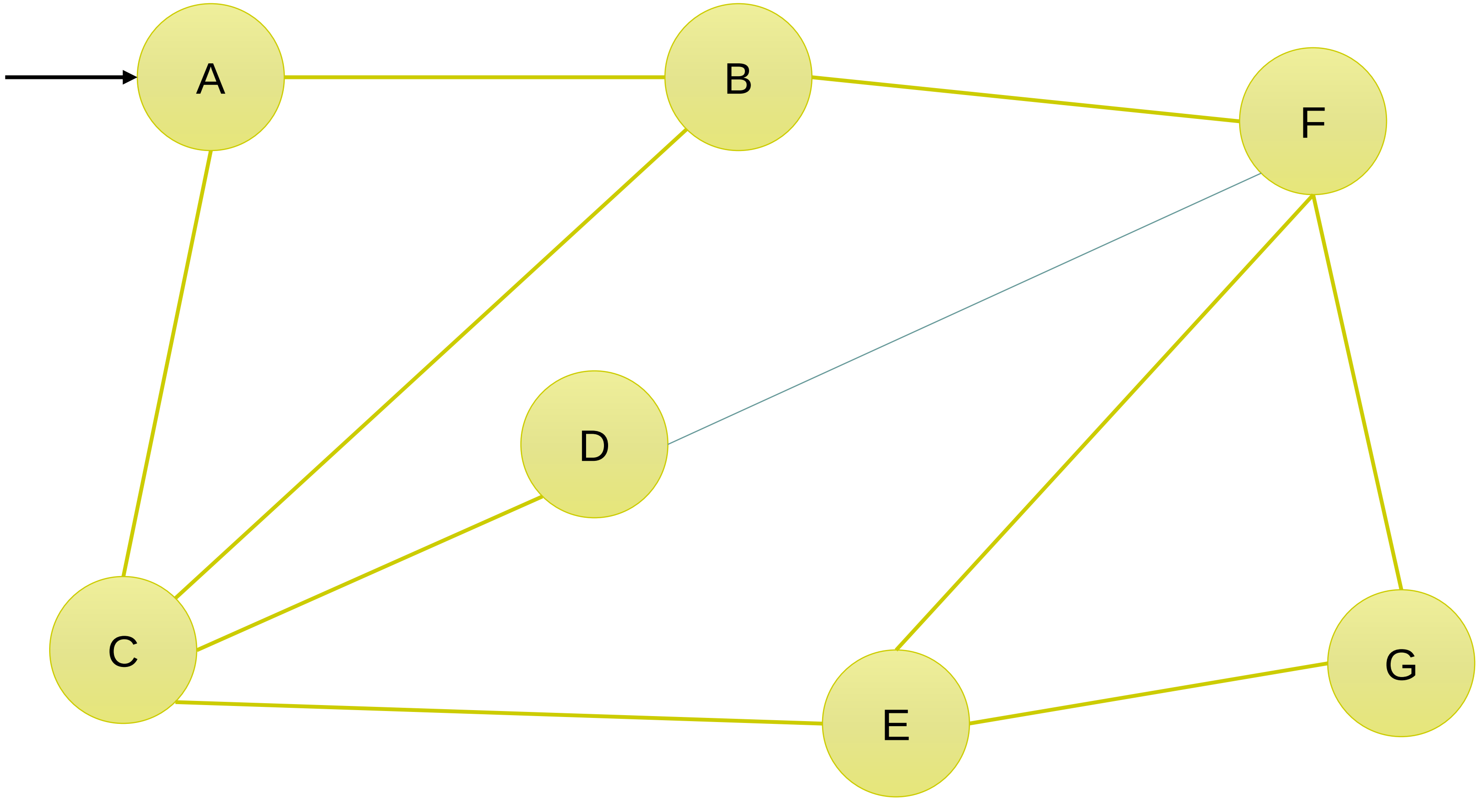
Depth-first Search (DFS)



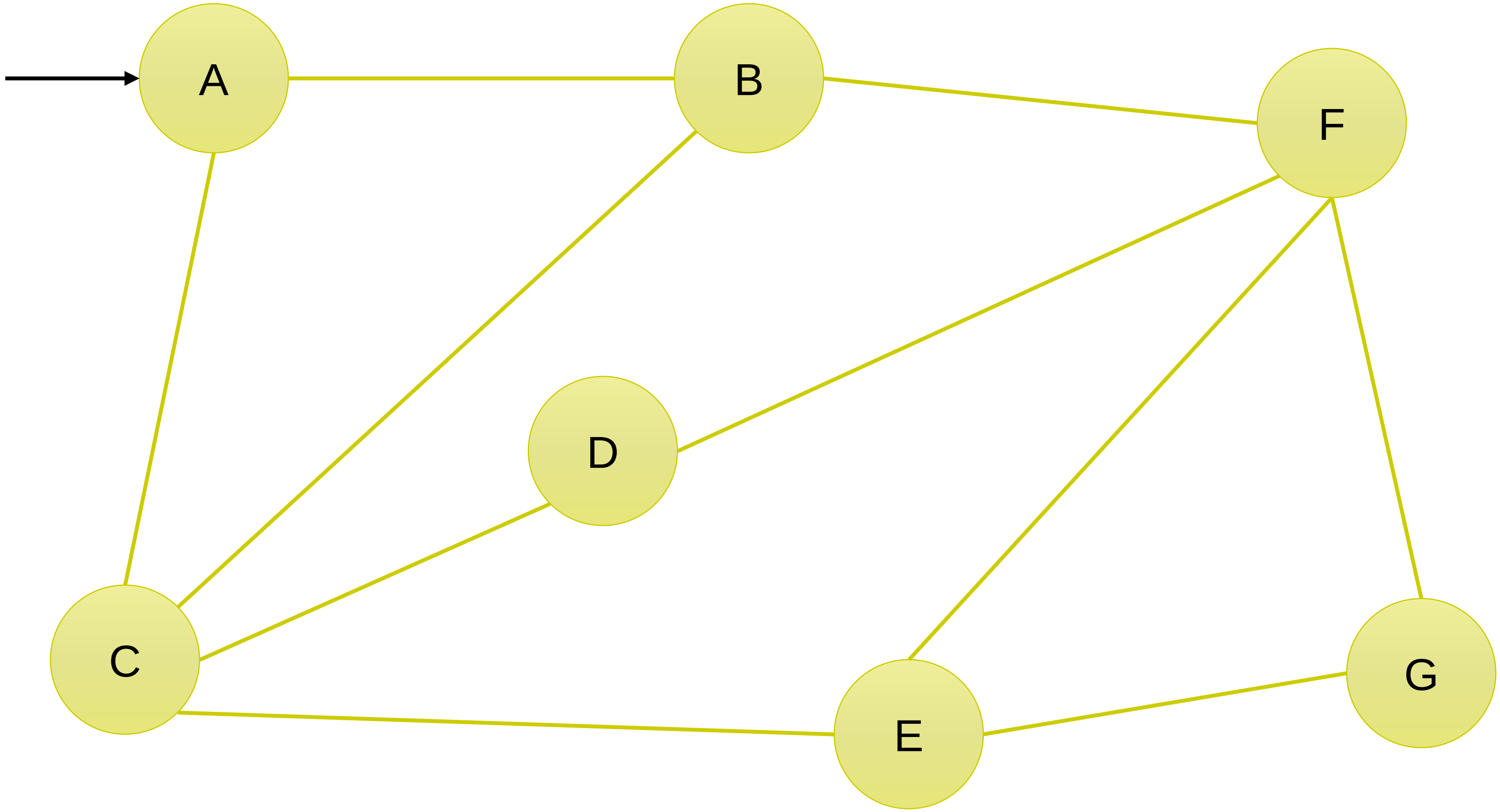
Depth-first Search (DFS)



Depth-first Search (DFS)



Depth-first Search (DFS)



Graph Traversals

```
GraphTraversal(G, v) {
  L ← An empty data structure
  L << v
  while (L is not empty) {
    x ← Remove next item from L
    Visit(x)
    for (each neighbor n of x) {
      L << n
    }
  }
}
```

How to make this generic code work as a BFS or DFS?