Depth-First Search (DFS)

- Another archetype for many important graph algorithms
- Methodically explore every vertex and every edge

Input: Given $G = (V, E)$

Output: (1) Two timestamps for every $v \in V$
- $d[v] = \text{when } v \text{ is first discovered.}$
- $f[v] = \text{when } v \text{ is finished.}$
(2) classification of edges
Depth-First Search (DFS)

- DFS idea:
  - go as far as possible, then “back up”
  - edges are explored out of the most recently discovered vertex $v$ that still have unexplored edges leaving
  - when all of $v$’s edges have been explored, the search “backtracks” to explore edges leaving the vertex from which $v$ was discovered.

- Three-color code for search status of vertices
  - White $=$ a vertex is undiscovered
  - Gray $=$ a vertex is discovered, but its processing is incomplete
  - Black $=$ a vertex is discovered, and its processing is complete
Depth-First Search (DFS)

DFS(G)  // main routine  :  DFS-Visit(u)  // subroutine
for each vertex u in V  :  color[u] = ‘‘gray’’
   color[u] = ‘‘white’’  :  time = time + 1
endif
endfor
for each vertex u in V  :  d[u] = time
   if color[u] = ‘‘white’’  :  for each v in Adj[u]
      DFS-Visit(u)  :  if color[v] = ‘‘white’’
      endif  :  DFS-visit(v)
   endif  :  endif
.endfor
// end of main routine  :  color[u] = ‘‘black’’
// end of subroutine
     time = time + 1
     f[u] = time
Depth-First Search (DFS)

Remarks:

▶ Vertices, from which exploration is incomplete, are processed in a LIFO stack.

▶ Running time: $\Theta(|V| + |E|)$

    not big-O since guaranteed to examine every vertex and edge.

▶ More properties of DFS, see pp.606-608 of [CLRS,3rd ed.]
Depth-First Search (DFS)

Classification of edges

- **T** = Tree edge = encounter new vertex (gray to white)
- **B** = Back edge = from descendant to ancestor (gray to gray)
- **F** = Forward edge = from ancestor to descendant (gray to black)
- **C** = Cross edge = any other edges (between trees and subtrees): (gray to black)

Note: In an undirected graph, there may be some ambiguity since edge $(u,v)$ and $(v,u)$ are the same edge. Classify by the first type that matches.
DFS vs. BFS

1. **DFS**: vertices from which the exploring is incomplete are processed in a LIFO order (stack)

   **BFS**: vertices to be explored are organized in a FIFO order (queue)

2. **DFS** contains two processing opportunities for each vertex \( v \), when it is “discovered” and when it is “finished”

   **BFS** contains only one processing opportunity for each vertex \( v \), and then it is dequeued