Graphs

Eugene Weinstein
New York University
Computer Science Department
Data Structures Fall 2008
Graph Definitions

• A graph $G = (V, E)$ is a set of nodes/vertices ($V$) and edges ($E$)

• Each edge connects a pair of vertices: $(u,v) \in E$

• In an undirected graph, edges simply associate two vertices

• In a directed graph, edges point from one vertex to another

• e.g., $(u,v)$ is an edge from $u$ to $v$
Graph Terminology

• A path is a sequence of vertices connected by edges

• A cycle is a path whose first and last vertices are the same and no vertex appears more than once

• A graph is connected if there is a path from any vertex to any other vertex
More Graph Terminology

- A weighted graph is one with weights (a.k.a. costs or lengths) on edges
Graphs and Trees

• What is the connection between graphs and trees?
• A tree is an acyclic connected graph
  • So graphs can be viewed as a generalization of trees
• Is this a tree?
  • Maybe it would help to redraw it?
Graphs and Trees

• What is the connection between graphs and trees?

• A tree is an acyclic connected graph
  • So graphs can be viewed as a generalization of trees

• Is this a tree?

• Maybe it would help to redraw it?
Graph Traversals

• We can traverse/scan a graph just as we would a tree
• But it is more tricky - what are the issues?
  • Cycles: we have to keep track of which vertices have been visited already
• We will discuss two traversals
  • Depth first search (DFS): generalization of inorder, preorder, postorder traversal
  • Breadth first search (BFS): generalization of tree version
Depth First Search

- Assume all vertices are are colored white/black, initially white

- DFSHelper(G, u)
  - u.color ← black
  - visit u  // Print or do other work
  - for each vertex v such that (u,v) is an edge in G
    - if (v.color = white), DFSHelper(G, v)

- DFS(G)
  - for each vertex v of G
    - if (v.color = white), DFSHelper(G, v)
DFS Properties

- Linear running time
  - Each node is colored once
  - Each edge is inspected once
  - $O(|V|+|E|)$ runtime complexity
- This is a tail-recursion implementation
  - i.e., we also use $O(|V|)$ call stack space
  - There is also a non-recursive implementation, using a stack
DFS Question

• For a tree, this corresponds to a pre-order traversal
  • How do we change it to post-order?

• DFSHelper(G, u)
  • u.color ← black
  • visit u  // Print or do other work
  • for each vertex v such that (u,v) is an edge in G
    • if (v.color = white), DFSHelper(G, v)
Breadth-First Search

- Input to BFS: graph G and starting vertex s, all initially white
- BFS(G,s)
  - s.color ← black
  - Q.enqueue(s)
  - while (!Q.empty)
    - u ← Q.dequeue
    - visit u
    - for each vertex v such that (u,v) is an edge in G
      - if (v.color = white)
        - v.color ← black
        - Q.enqueue(v)
BFS Properties

• Linear runtime complexity

• Each vertex is visited at most once

• Each edge is inspected at most once

• Runtime: $O(|V| + |E|)$