Graphs

University of Massachusetts Amherst
ECE 242 – Data Structures and Algorithms
Lecture 22

Graph data structures

- Graph components
  - Vertices (nodes)
  - Edges
- Allows more general relationships between nodes
  - Arbitrary number of edges
Graph terminology

• Graph types
  – Undirected edges vs. directed edges
  – Unweighted edges vs. weighted edges

• Adjacency
  – Two vertices adjacent iff connected by single edge

• Path
  – Sequence of edges

• Connected graph
  – A path exists between all pairs of vertices

Graph examples

• Geographic data
  – Vertices: locations
  – Edges: roads

• Social network
  – Vertices: persons
  – Edges: personal/professional connection

• Internet
  – Vertices: computers/routers
  – Edges: data links
Graph implementation

• Vertex
  
  ```java
  public class Vertex {
    public String name;
    public int graphIndex; // index of adjacency matrix position of node
    public boolean visited;

    public Vertex (String s) {
      name = s;
      graphIndex = -1; // invalid position by default
      visited = false;
    }

    public String toString() {
      return name;
    }
  }
  ```

• Graph
  
  ```java
  public class Graph {
    private int maxVertices;
    private Vertex[] vertices; // array of nodes
    private int[][] edges; // adjacency matrix
    int activeVertices;

    public Graph(int maxSize) {
      maxVertices = maxSize;
      vertices = new Vertex[maxVertices];
      edges = new int[maxVertices][maxVertices]; // adjacency matrix
      activeVertices = 0;
    }

    ...
  }
  ```
Adding a vertex

```java
public void addVertex(Vertex v) {
    if (activeVertices >= maxVertices) {
        System.out.println("Graph full");
        return;
    }
    vertices[activeVertices] = v;
    // add vertex to list of vertices
    v.graphIndex = activeVertices;
    // record index of vertex in graph
    activeVertices++;
    // increment vertex count
}
```

Adding an edge

- Adjacency matrix indicates existence of edge
  - ‘0’ indicates no edge
  - ‘1’ indicates edge
  - Direction does not matter (yet)

```java
public void addEdge(Vertex v1, Vertex v2) {
    edges[v1.graphIndex][v2.graphIndex] = 1;
    edges[v2.graphIndex][v1.graphIndex] = 1;
}
```
Operations on graph

- Graph has less structure than tree
  - Need more general operations
- Traversal / search
  - Find node by following edges from start node
  - Traversal methods
    - Depth-first traversal
    - Breadth-first traversal

Next Steps

- Lecture on Wednesday