Introduction to Graphs

15-121

Introduction to Data Structures

Ananda Gunawardena
Graphs are everywhere
An Airline route Map
Finding the Shortest Path
Lots of applications
Many real world problems can be modeled using graphs

- **Airline Route Map**
  - What is the fastest way to get from Pittsburgh to St Louis?
  - What is the cheapest way to get from Pittsburgh to St Louis?

- **Electric Circuits**
  - Circuit elements - transistors, resistors, capacitors
  - is everything connected together?
    - Depends on interconnections (wires)
  - If this circuit is built will it work?
    - Depends on wires and objects they connect.
Graph Definitions

• **Graph**
  - A set of vertices (nodes) \( V = \{v_1, v_2, ..., v_n\} \)
  - A set of edges (arcs) that connects the vertices \( E = \{e_1, e_2, ..., e_m\} \)
  - Each edge \( e_i \) is a pair \((v, w)\) where \( v, w \) in \( V \)
  - \(|V| = \) number of vertices (cardinality)
  - \(|E| = \) number of edges

• **Graphs can be**
  - directed (order \((v,w)\) matters)
  - Undirected (order of \((v,w)\) doesn’t matter)

• **Edges can be**
  - weighted (cost associated with the edge)
  - eg: Neural Network, airline route map (vanguard airlines)
Graph Representations
Graph Representation

- How do we represent a graph internally?
- Two ways
  - adjacency matrix
  - Adjacency list
- Adjacency Matrix
  - Use matrix entries to represent edges in the graph
- Adjacency List
  - Use an array of lists to represent edges in the graph (we will discuss this later)
Adjacency Matrix

- Adjacency Matrix
  - For each edge \((v,w)\) in \(E\), set \(A[v][w] = \text{edge\_cost}\)
  - Non existent edges with logical infinity
- Cost of implementation
  - \(O(|V|^2)\) time for initialization
  - \(O(|V|^2)\) space
    - ok for dense graphs
    - unacceptable for sparse graphs
Adjacency List

- Adjacency List
  - Ideal solution for sparse graphs
  - For each vertex keep a list of all adjacent vertices
  - Adjacent vertices are the vertices that are connected to the vertex directly by an edge.
  - Example

List 0

```
1 → 2
```

List 1

```
2 → 0 → 1
```

List 2

```
1
```
Adjacency List

- The number of list nodes equals to number of edges
  - $O(|E|)$ space
- Space is also required to store the lists
  - $O(|V|)$ for $|V|$ lists
- Note that the number of edges is at least $\text{round}(|V|/2)$
  - assuming each vertex is in some edge
  - Therefore disregard any $O(|V|)$ term when $O(|E|)$ is present
- Adjacency list can be constructed in linear time (wrt to edges)
Breadth First Traversal

- Algorithm
  - Start from any node in the graph
  - Traverse to its neighbors (nodes that are directly connected to it) using some heuristic
  - Next traverse the neighbors of the neighbors etc.. Until some limit is reach or all the nodes in the graph are visited
  - Use a queue to perform the breadth first traversal
Depth First Traversal

- **Algorithm**
  - Start from any node in the graph
  - Traverse deeper and deeper until dead end
  - Back track and traverse other nodes that are not visited
  - Use a stack to perform the depth first traversal
Next: Graph Algorithms