Graphs

Week 7

Required Activities

- Check Announcements regularly (every 2-3 days)
- Read DSA book: Chapter 9
- Review supplemental materials – Graph ADT
- Keep working on Assignment 3 due 12/21

Graph

- Non-linear data structure that represents relationship between pairs of elements not necessary hierarchical
- **Terminology:**
  - **vertices** – a set of nodes
  - **edge** – connection between two vertices
    - **directed edge:** first vertex is origin and second is destination with edge pointing to destination (e.g. one way road traffic)
    - **undirected edge:** unordered pair of vertices (e.g. railroad lines)
  - **directed graph** – all edges are directed (e.g. route network)
  - **undirected graph** – all edges are undirected (e.g. flight network)
  - **tree** – a graph with no cycles
  - **self loop** – edge that connects vertex to itself
  - **path** – sequence of vertex and edges where no vertex/edge repeats
  - **connected graph** – can reach all nodes from all nodes
  - **disconnected graph** – it is not possible to reach some nodes from other nodes
  - **weighted graph** – each edge has some data assigned to it

Uses of Graph

- Represent components in electronic circuits
- All networks whether physical (e.g. highway) or virtual (e.g. local area network)
- Chemical compound representation
- Social network representation

Graph Representation

Can be represented:

- **Adjacency matrix** – sequence matrix with row-column for each vertex. Value of 1 signifies edge and value 0 means there is no edge. For example if we have V1 and V2 vertices with edge from V1 and V2 and V2 and V1 then we would have 1 in [V1][V2] and [V2][V1] locations.
- **Incidence matrix** –sequence matrix that has row for every vertex and column for every edge. Value 0 means there is no edge connection for that vertex, 1 means the edge direction is from the vertex, and -1 means edge direction is to the vertex.
- **Adjacency list** – linked list representation where node has vertex number and reference to the node it connects to. Each vertex has its own linked list. So you can have array list where each element has a linked list for that vertex.

Graph Traversal

Graph traversal is also called graph search. It starts at any vertex, goes through edges, and marks vertices that it visited. There are two algorithms that are typically used:

- **Depth First Search (DFS):** similar to pre-order for a tree. It can use a stack. Basic steps include:
  - Start at some vertex v1, considers all connected vertices v2..vn, visit vertex at v2 and all vertices it is connected to
  - Once it cannot go any further, it backtracks to v1 and goes to next vertex it is connected to that it has not visited yet
  - Keeps backtracking until visits all vertices v1 is connected to
  - Repeats the above for all other vertices that did not visit yet

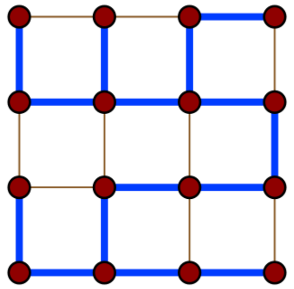
- **Breath First Search (BFS):** similar to level-order traversal for a tree. It can use a queue. Basic steps include:
  - Start at some vertex v1, level 0. And visits all vertices at level 1 (vertices of distance 1 from starting vertex v1) which are basically all vertices it is connected to
  - Then it visits all vertices at level 2, that it has not visited, which are the vertices that are connected to the vertices v1 is connected to
  - Repeats until all levels are visited

DFS vs BFS

- DFS has lower memory requirements as it does not need to store child pointers at each level
- DFS has pre-order traversal and BFS has level-order traversal
- Which is faster will depend on the problem we are trying to solve. E.g. if data at lower level, BFS may be faster and if data is at maximum depth, DFS may be faster.

Spanning Tree

- **Spanning tree T** of undirected graph G is a subgraph where it has all vertices of G and is a tree with minimum number of edges. A graph may have multiple spanning trees. It is always a connected graph and it cannot contain a cycle. For example, the below image is a spanning tree (blue heavy edges) of a grid graph



- **Minimum Spanning Tree (MST) –** The cost of spanning tree is the sum of all weights of its edges so MST is where the cost is as small as possible

- **Algorithms using spanning tree:**
  - Dijkstra's algorithm
  - A* search algorithm
  - Prim's algorithm

Image Credit:By David Eppstein - Own work, Public Domain, https://commons.wikimedia.org/w/index.php?curid=2671392

Questions ?

- Post in the discussions
- Send email to RMcFadden@HarrisburgU.edu
- Respond usually within 48hours