Lecture 37

Graphs

Graphs are very flexible data structures that have many different applications.

Where do graphs come up in real life?

- transportation networks (flights, roads, etc.)
  - flights and flight patterns.
  - what sort of questions might we ask? What sort of application might we be interested in having a graph?
    - booking flights, picking shortest time? shortest distance?
    - airlines save fuel, number of people who use the route
  - google maps
    - driving directions, mapping out sightseeing
- communications networks/utility networks
  - electrical grid, phone networks, computer networks
  - minimize cost for building infrastructure
  - minimize losses, route packets faster
- social networks
  - does this person know that person.
  - Can this person introduce me to that person – job opportunities

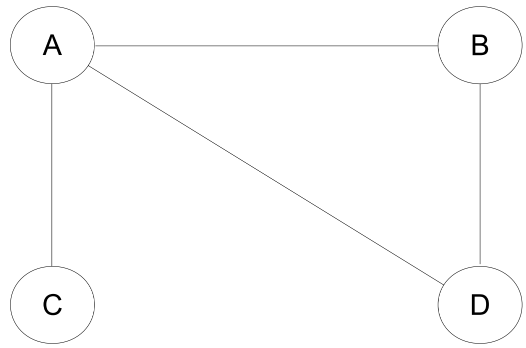
Definitions

Formally, we can define a graph this way:

A graph G = (V, E), where V is a finite, non-empty set of vertices and E is a binary relation on V (that is, E is a set of edges that connects pairs of vertices).

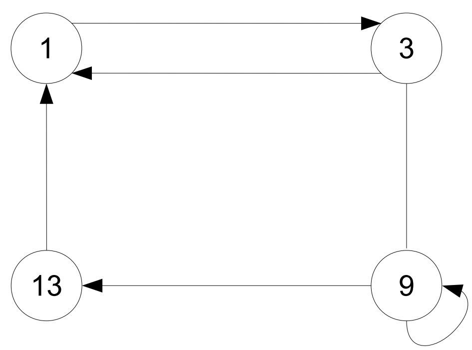
In general, there are two major types of graphs – Directed and Undirected.

Undirected graphs can be described as:




```
G = (V, E)
V = {A, B, C, D}
E = {{A,C}, {A,B}, {A,D}, {B, D}}
```

Directed graphs can be described as:




```
G = (V, E)
V = {1, 3, 9, 13}
E = {(1,3), (3,1), (13, 1), (9,13), (9,9)}
```

path – a sequence of connected vertices.

simple path – a path where all vertices occur only once.

path length – number of edges in the path. Example from undirected graph above, path C-A-D-B has length 3.

cycle – path of length ≥ 1 that begins and ends with the same vertex. From above, path A-D-B-A is a cycle.

simple cycle – a simple path that begins and ends with the same vertex. (Note, we do not count the start and end vertex twice when determining a simple path). From above, A-D-B-A is also a simple cycle.

self loop A cycle consisting of one edge and one vertex. Typically, we talk about self loops in directed graphs. See vertex 9 above.

incident edges:

Edge (A,B) above is **incident on** A and B. Thus, A and B are said to be **adjacent**.

Edge (13, 1) above is **incident from** vertex 13, and **incident to** vertex 1. 13 and 1 are also said to be **adjacent**.

degree – number of incident edges for a vertex. From above, vertex A has degree 3. Vertex 9 has degree 2.

simple graph – A graph with no self loops.

acyclic graph – a graph with no cycles.

connected graph – a graph where every pair of vertices is connected by a path.

strongly connected graph – a connected, directed graph.

weakly connected graph – a directed graph that would be connected if all of its directed edges were replaced by undirected edges.

forest – an acyclic, undirected graph.

tree – a connected, acyclic, undirected graph.

Graph representations

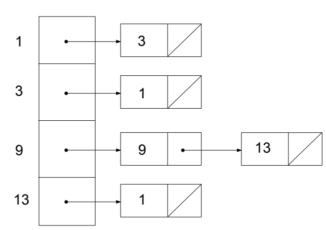
There are two basic types of graph representation, though variations of both are common.

1. Adjacency Matrix (e.g., for undirected graph above)

	A	B	C	D
A	0	1	1	1
B	1	0	0	0
C	1	0	0	1
D	1	0	1	0

   Good for dense graphs, and have constant time lookup of edges.

2. Adjacency List (e.g., for directed graph above)



   Good for sparse graphs, have linear time lookup of edges.