COMP 282
Advanced Data Structures

Lecture 03

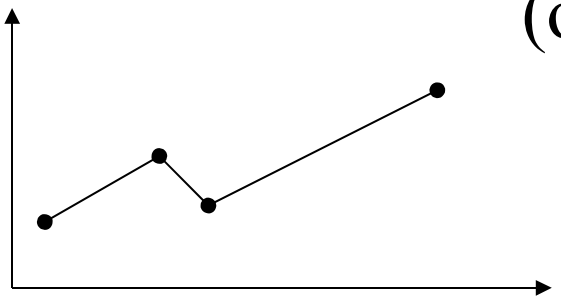
Graphs

Introduction and Terminology

Graphs

- We are not talking about classic line “graphs”.

  (or skin grafts, ewww!)



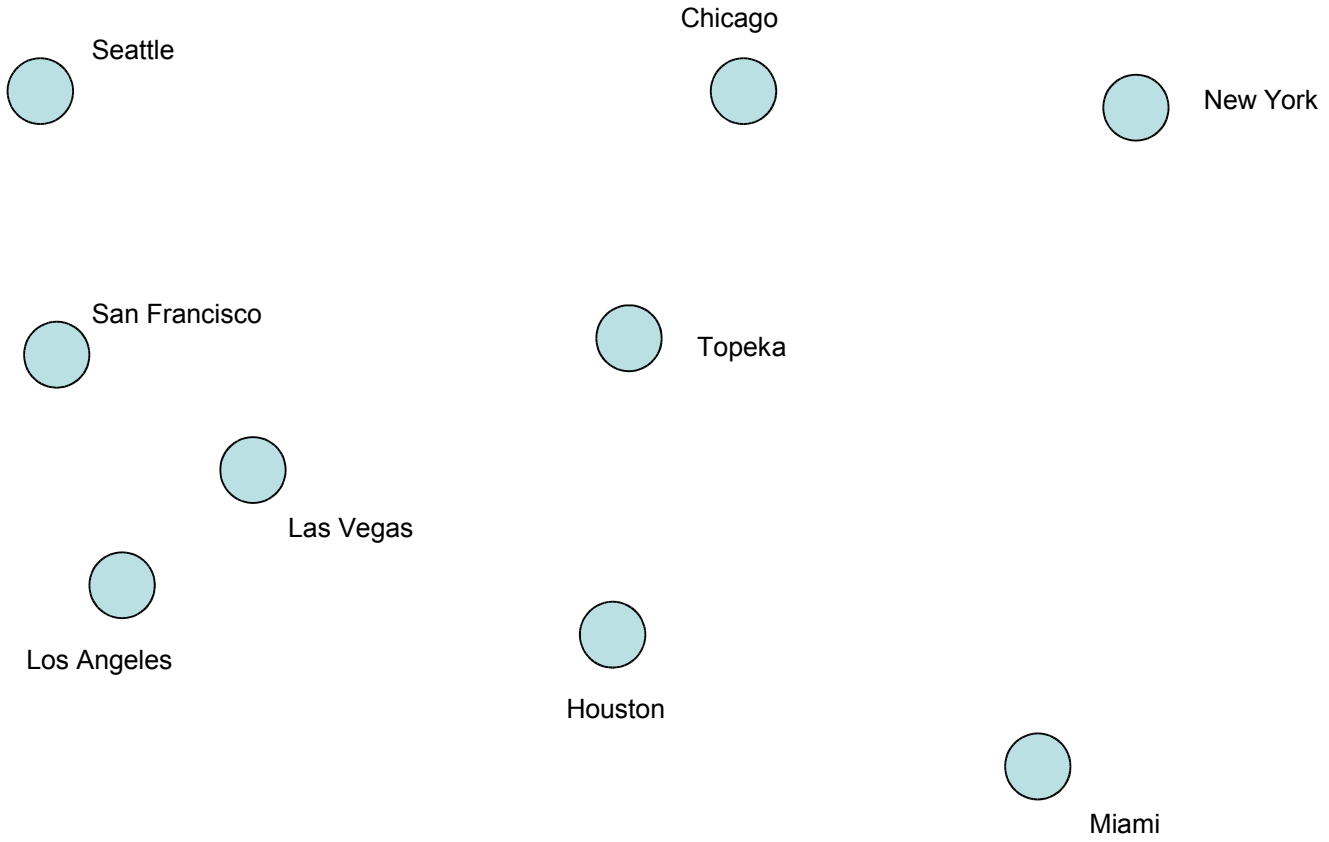
- Though some concepts from line graphs do provide some understanding of generalized graphs.
  - Graphs are used to represent data
  - Graphs illustrate relationships of data points.

Graph Theory

- The concept of “graph” as it pertains to computer science is defined by two sets.
  - The set ***V***; the vertices, or “nodes” of the graph.
  - The set ***E***; the edges that connect the vertices.
- Line graphs are a subset of the more general class of graphs as defined here.

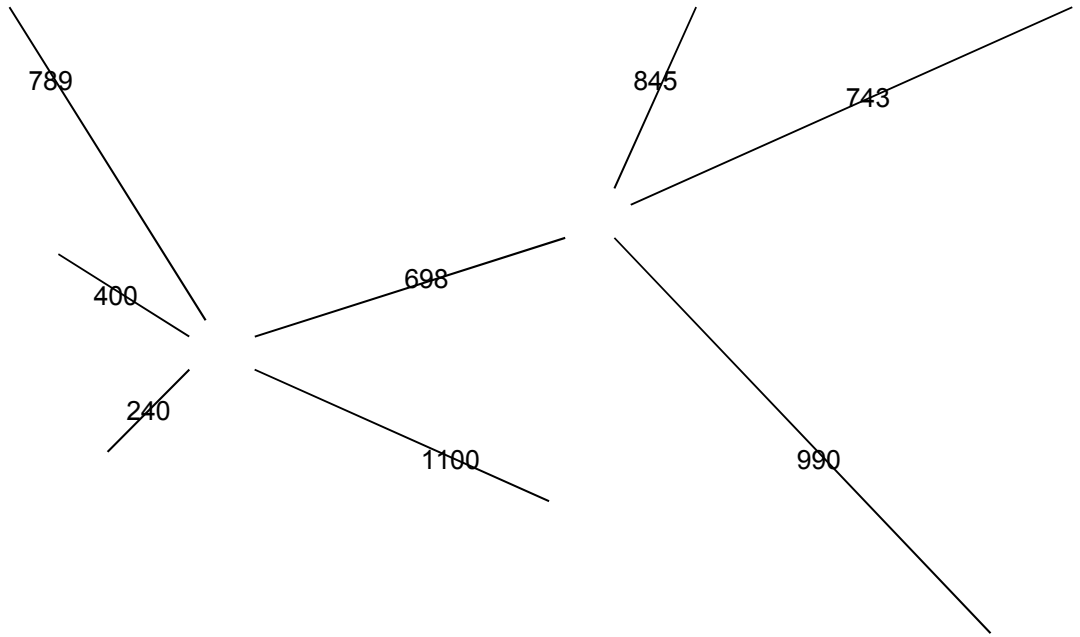
Vertex set example

- Here's an example of a set of vertices…



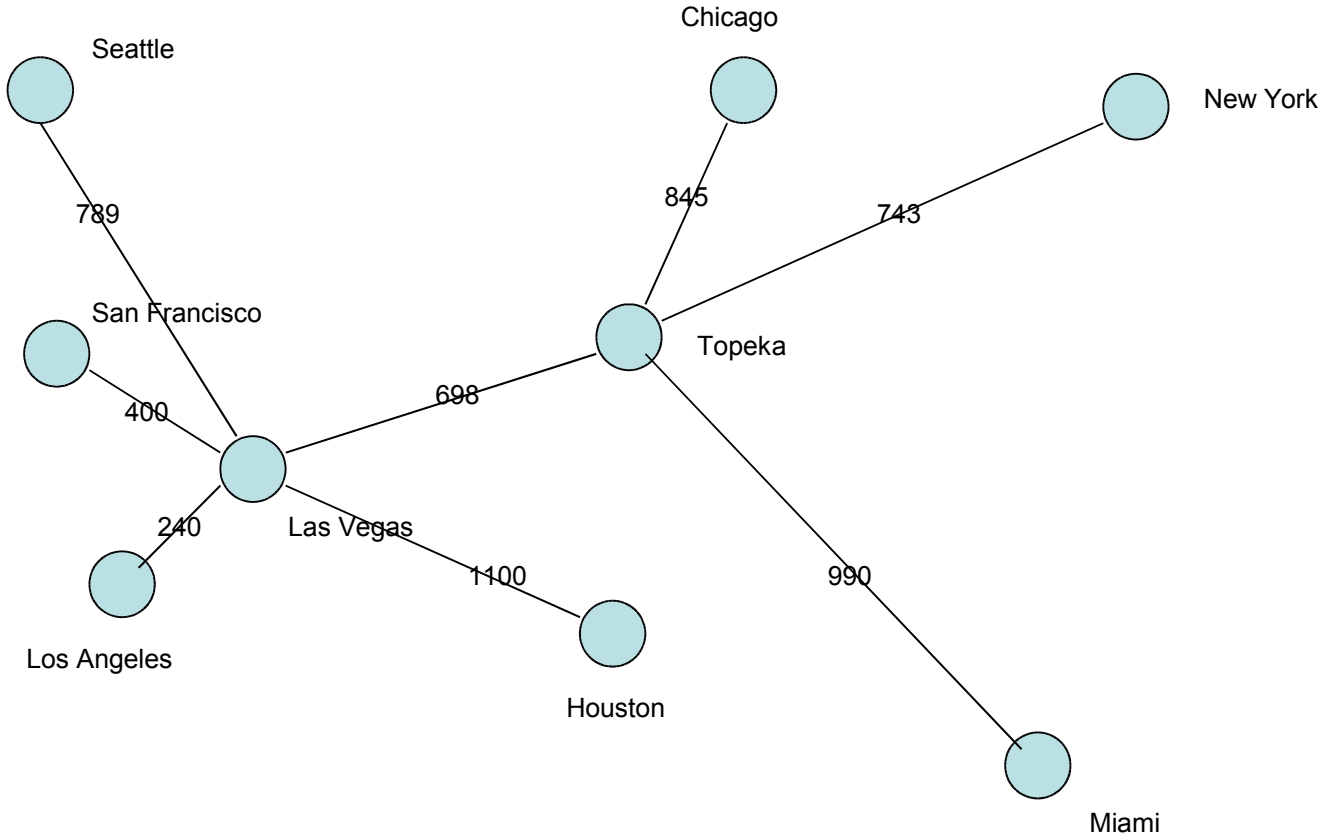
Edge set example

- Here's an example of a set of edges…



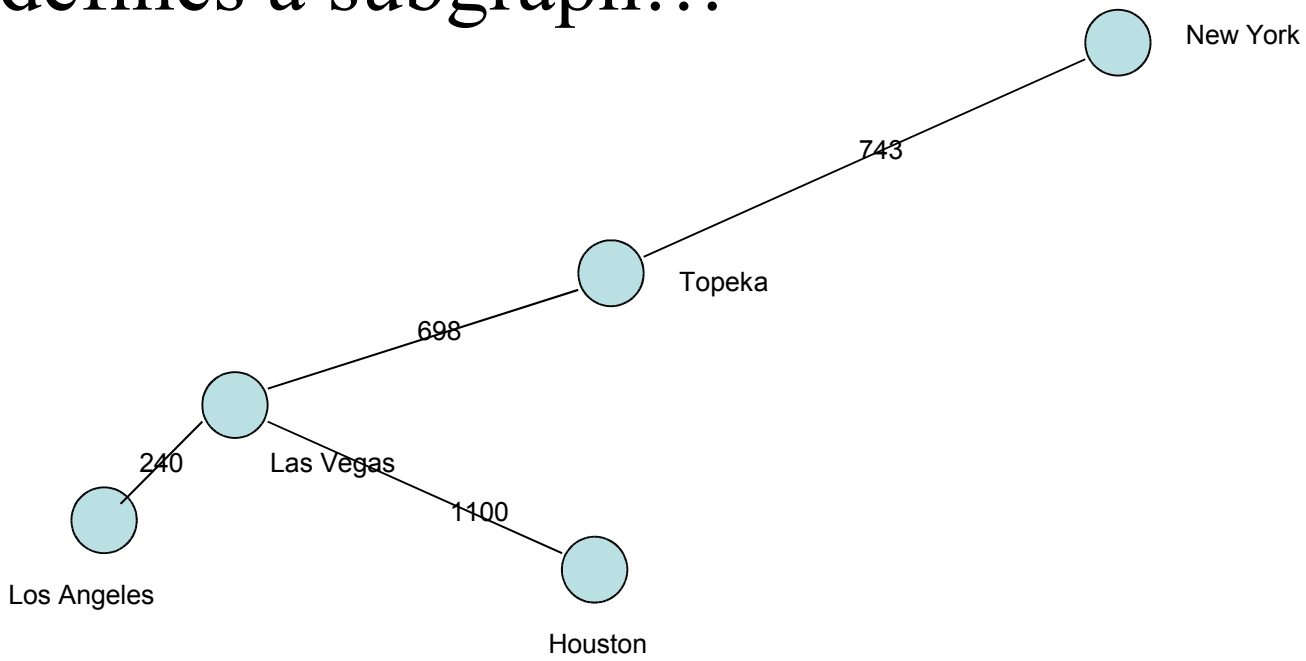
Graph Example

- The two sets define this graph…



subgraphs

- A subset of vertices with a subset of edges defines a subgraph…

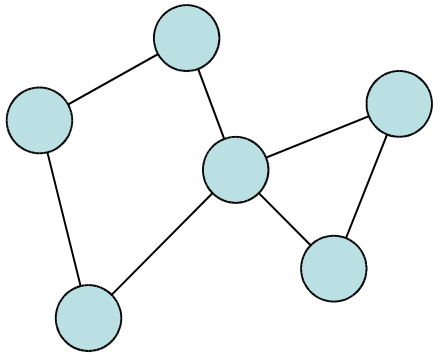


Terminology

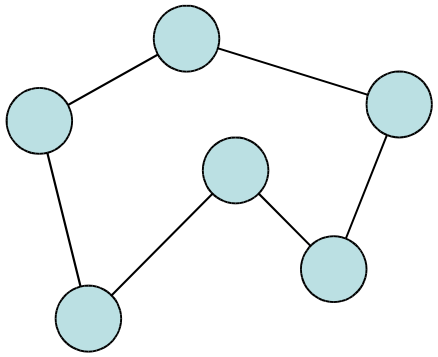
- Adjacent: Two nodes in a graph are considered to be “adjacent” if they are joined by an edge.
- Path: a path between two vertices is a sequence of edges that begin at the first vertex and end at the other.
- Simple Path: A path that does not go through the same vertex more than once.

Terminology Cont.

- Cycle: A path that begins and ends at the same vertex.
- Simple Cycle: a cycle that does not pass through other vertices more than once.



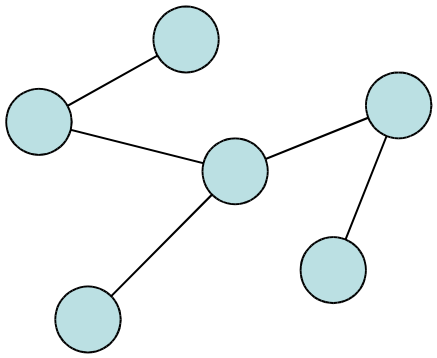
Cycle



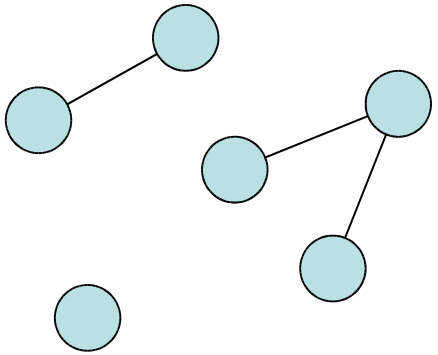
Simple Cycle

Connected / Disconnected

- Connected: A graph is considered connected if there exists a path from every vertex to every other vertex
- Disconnected: there exists a set of nodes that cannot be reached from some other set of nodes.



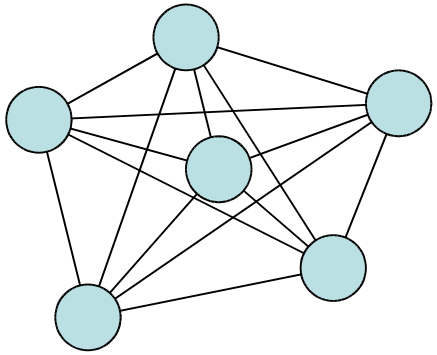
This graph is connected.



disconnected

Complete

- Complete: A graph is complete if there exists an edge between every pair of nodes.



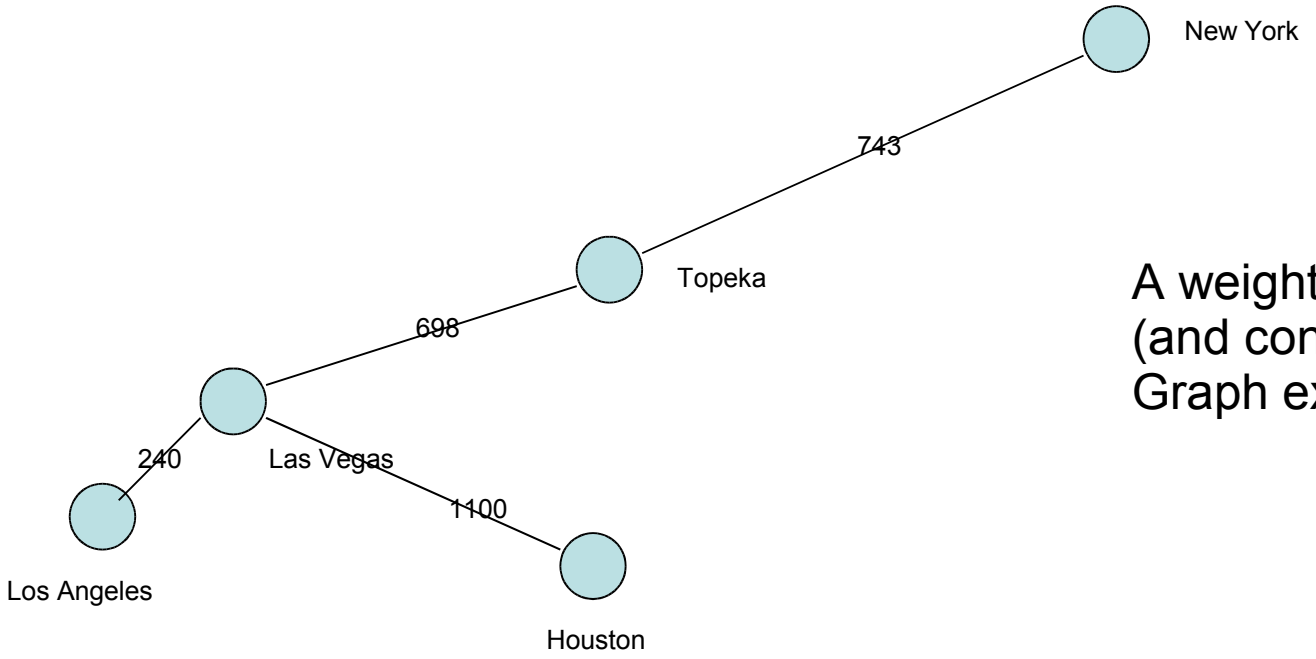
This graph is complete.

Self Edges

- Graphs cannot have “self edges”. That is there cannot be an edge from a node to itself.
  - This constraints is included in the Carrano-Prichard book.
  - Finite state machines are represented as graphs and often contain self edges.

Weighted Graphs

- Edges can be labeled. Graphs with labeled edges are called “weighted graphs”.



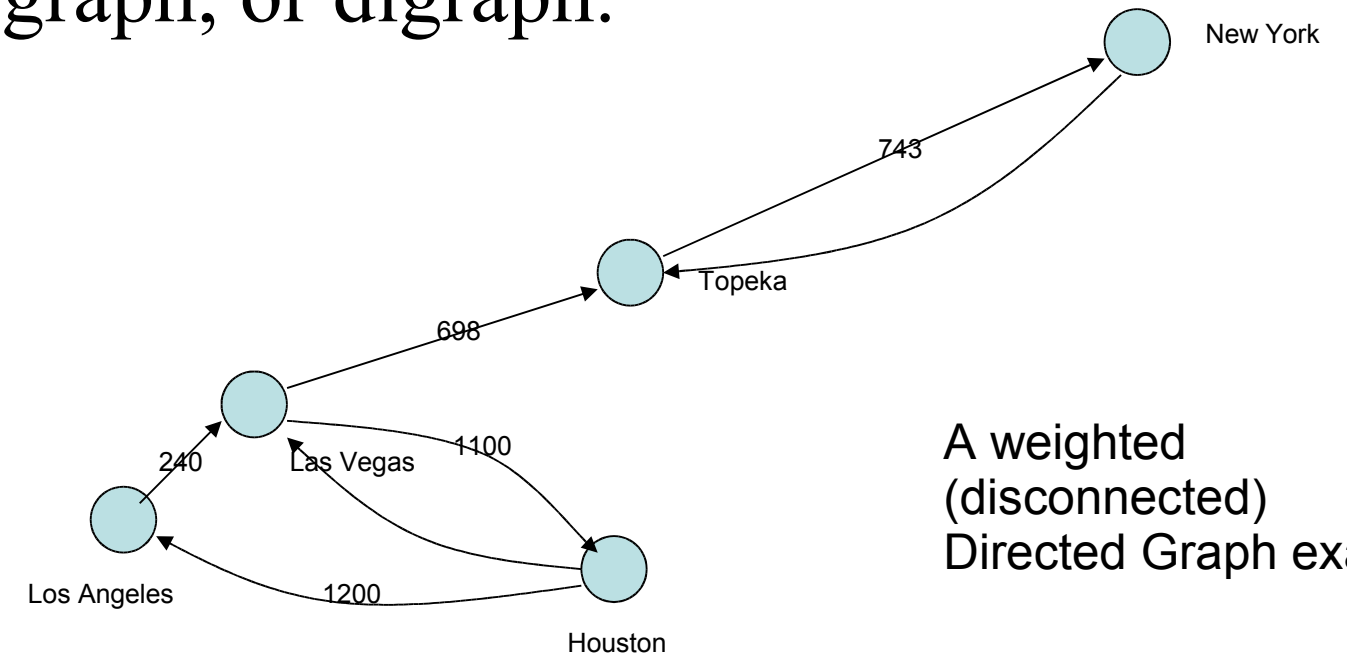
A weighted (and connected) Graph example.

Undirected Graphs

- All examples so far have been examples of “undirected” graphs.
- In an undirected graph the edges between vertices are valid in either direction. (If you can get to node B from node A using edge E then you can get from A from B using the same edge E.

Directed Graphs

- Directional graphs: If valid directions are imposed on edges then the graph is a directional graph, or digraph.



A weighted (disconnected) Directed Graph example.

Graphs as ADTs

- Operations:
  - Create empty graph
  - Determine if graph is empty
  - Determine number of vertices
  - Determine number of edges
  - Determine whether an edge exists between two vertices; return weighted value
  - Insert a new, unique vertex in the graph
  - Delete a particular vertex from a graph and any edges between the deleted vertex and any other vertex.
  - Insert a new, unique edge between two vertices
  - Delete an edge from the graph
  - Retrieve a vertex from the graph.