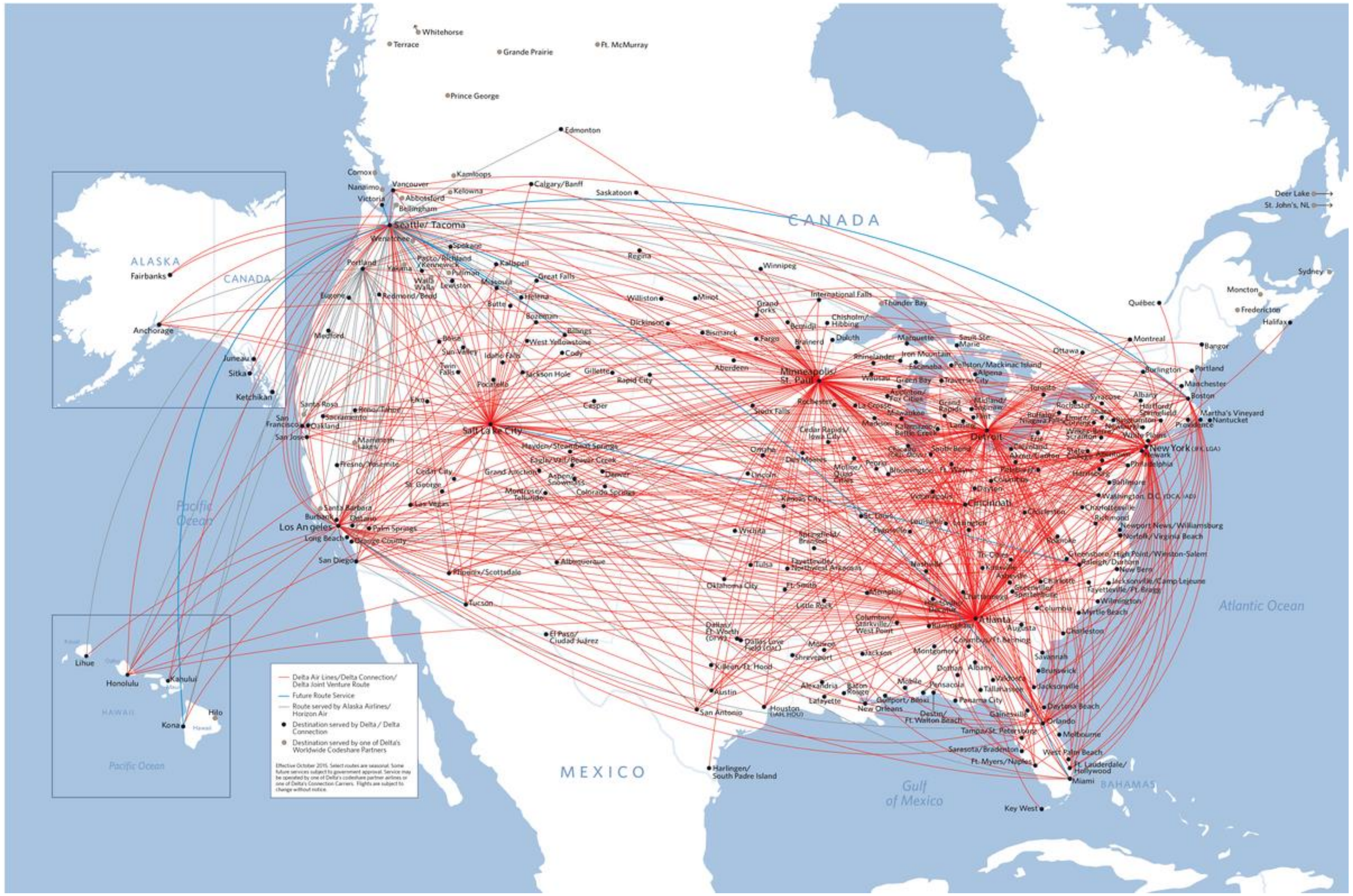
Graphs:
Introduction and Representation

Outline

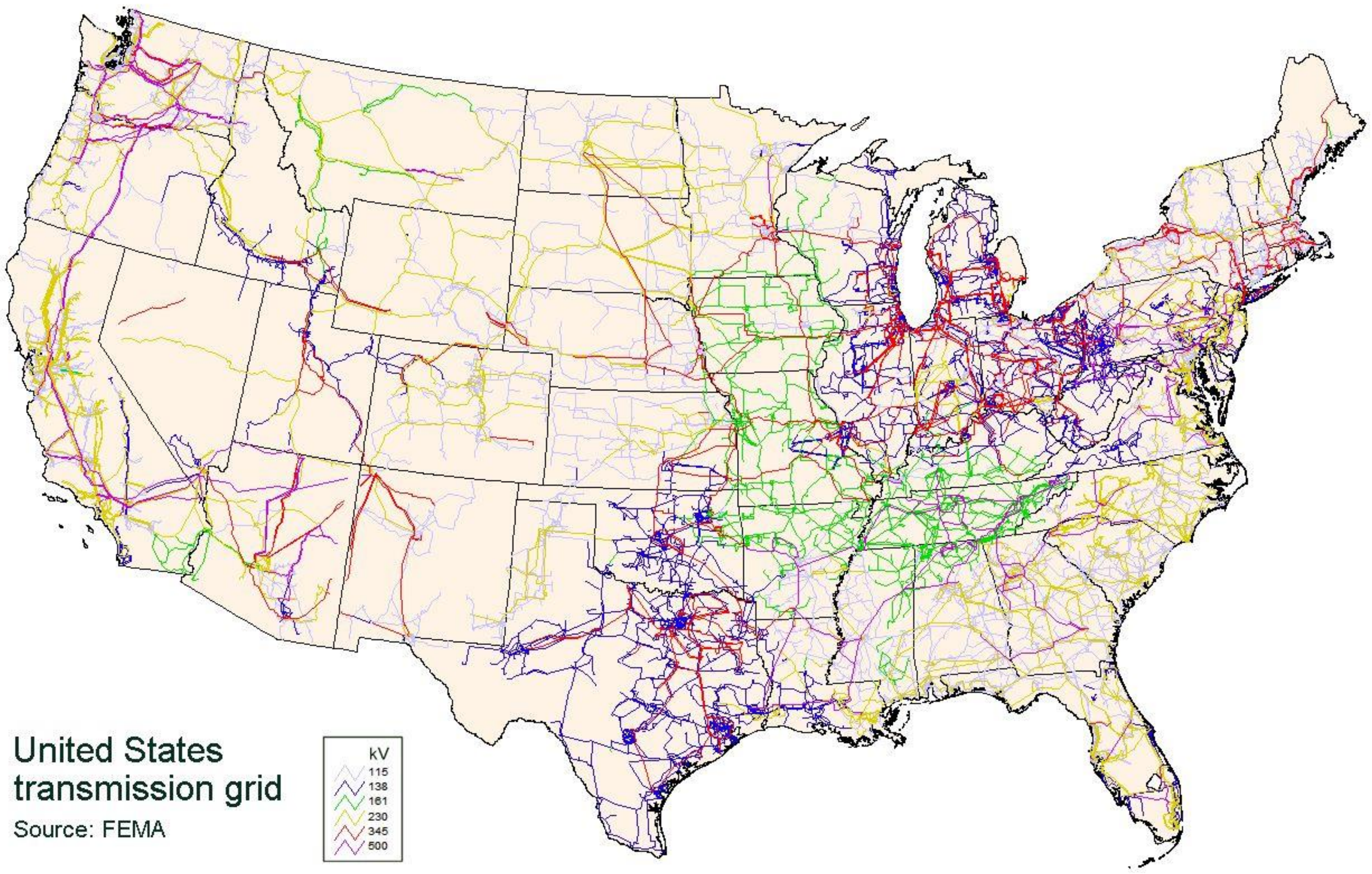
- Applications
- Definitions
- Matrix Representation
  - Directed Graphs
  - Undirected Graphs
  - Pros and Cons
- Linked List Representation
  - Implementation

Airline Routes



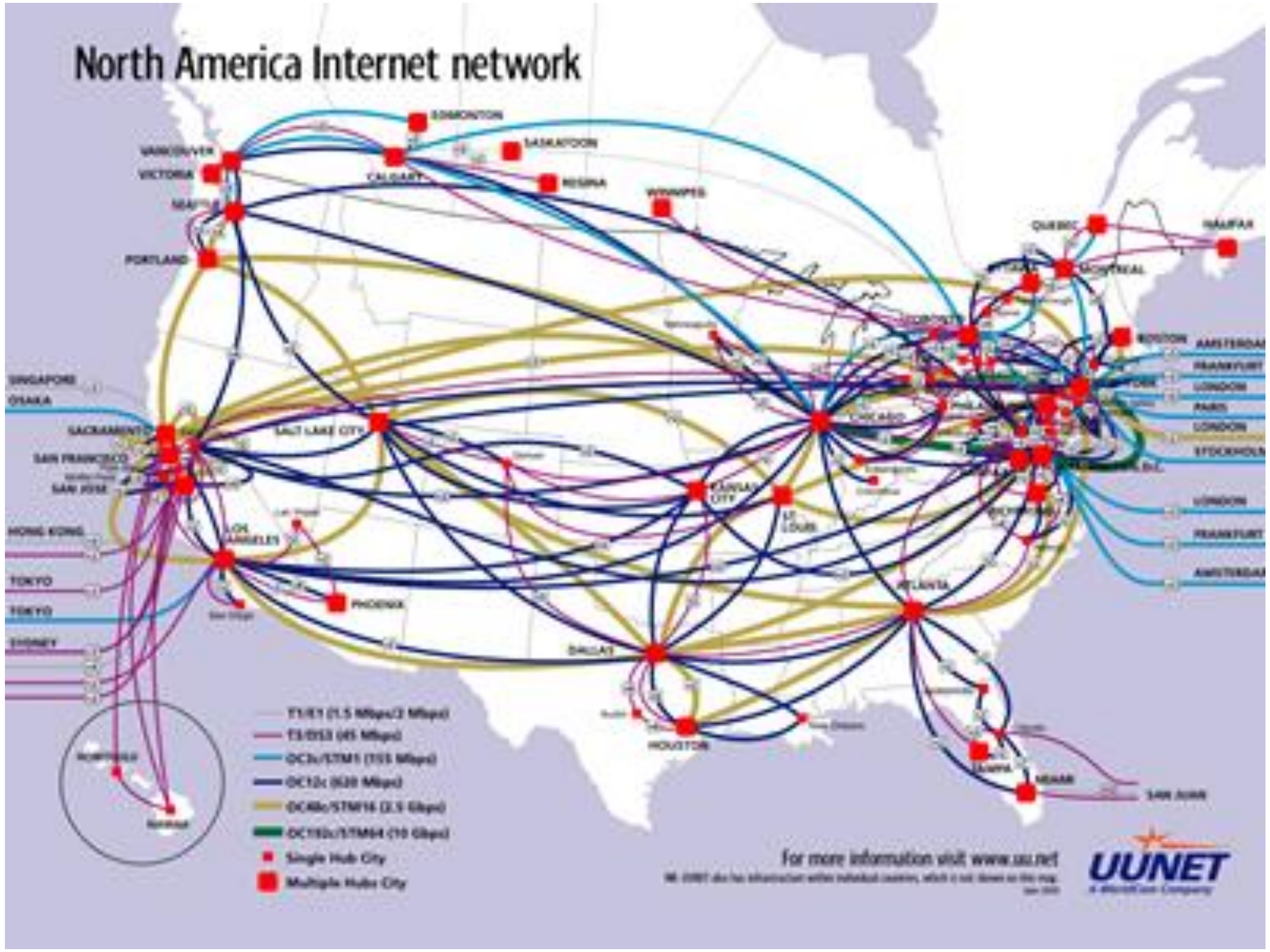
https://news.delta.com/route-map-us-canada

Electric Power Grid



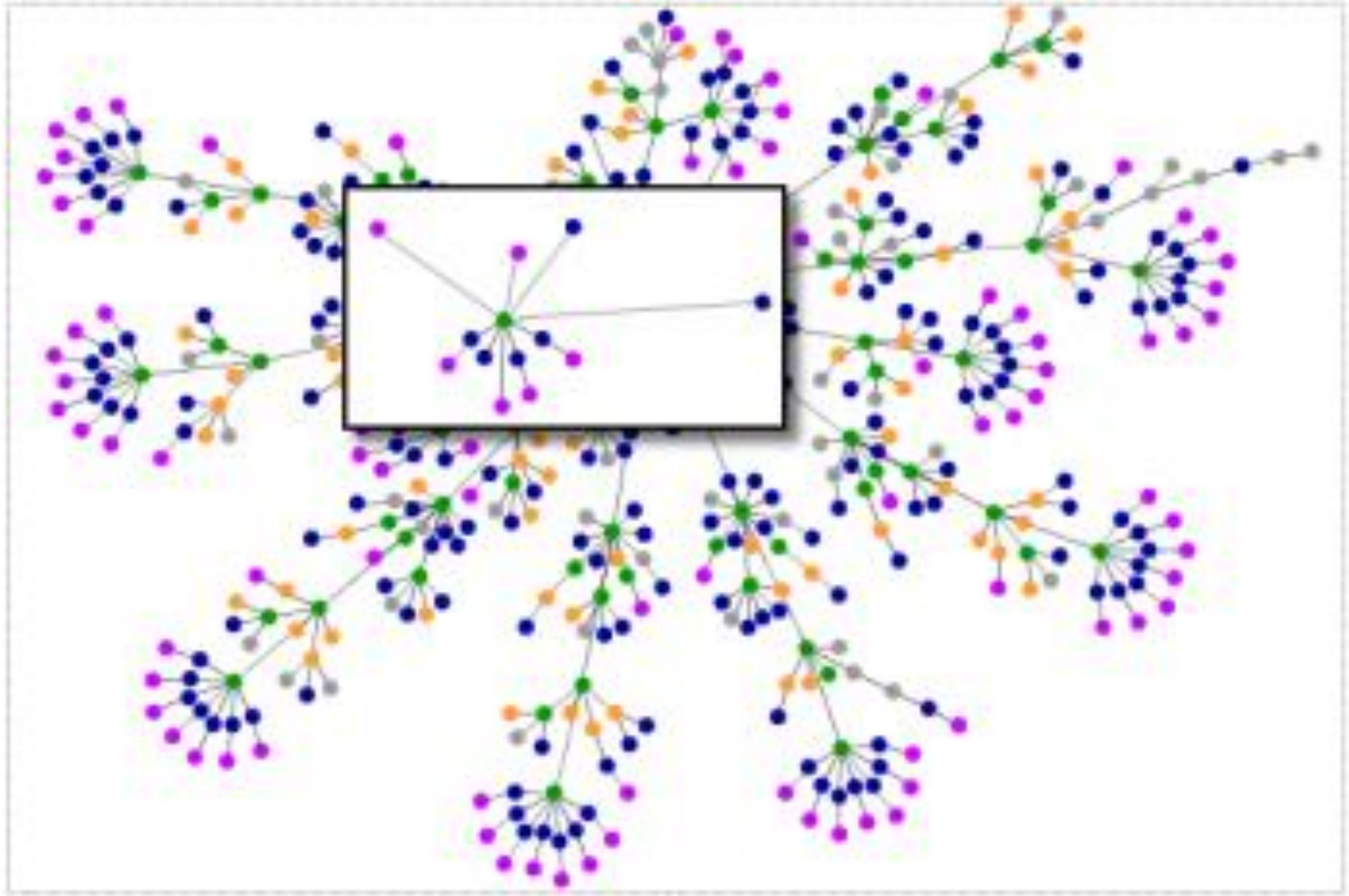
http://www.geni.org/globalenergy/library/national_energy_grid/united-states-of-america/americannationalelectricitygrid.shtml

Physical Topology of the Internet



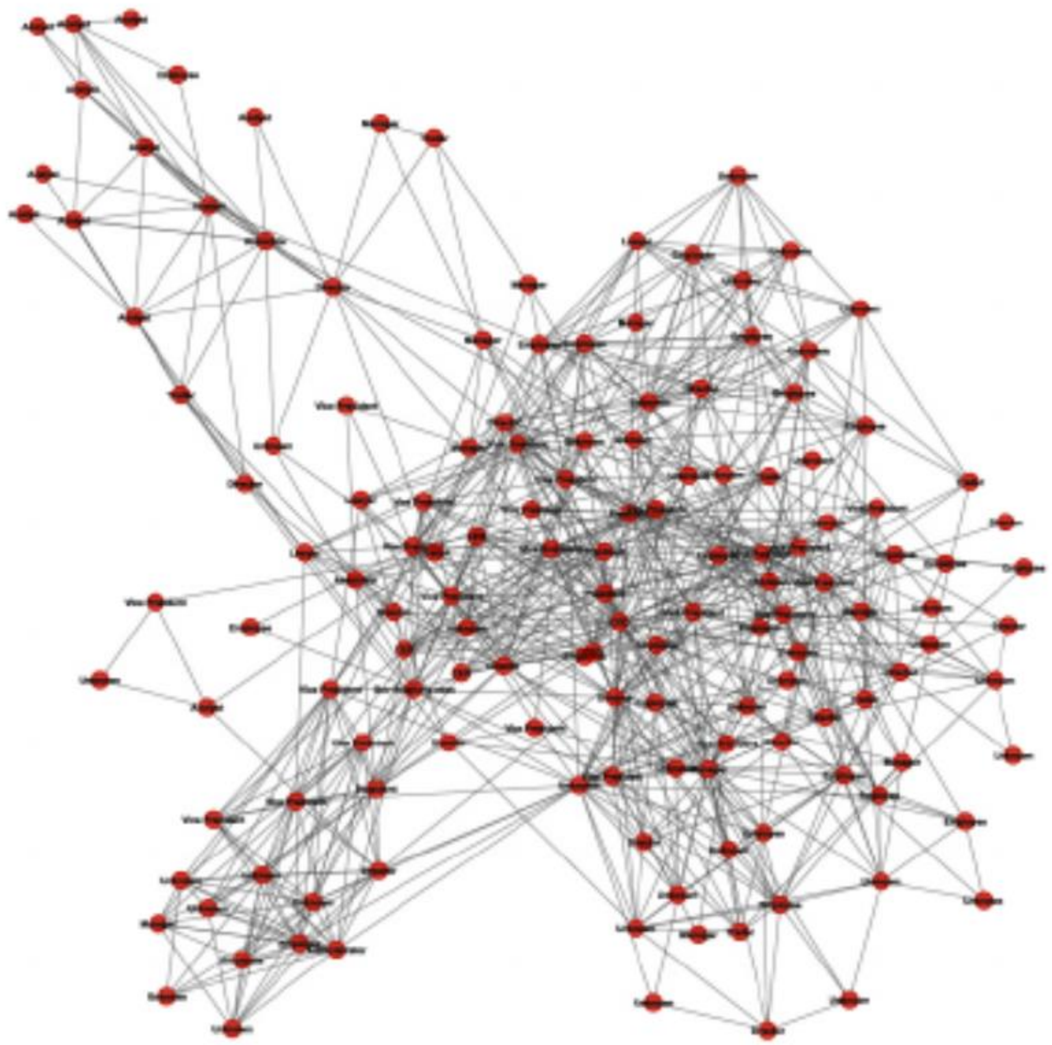
http://www.visualcomplexity.com/vc/images/270_big01.jpg

Website Linking



https://www.toprankblog.com/2009/03/link-building-tips-new-websites/

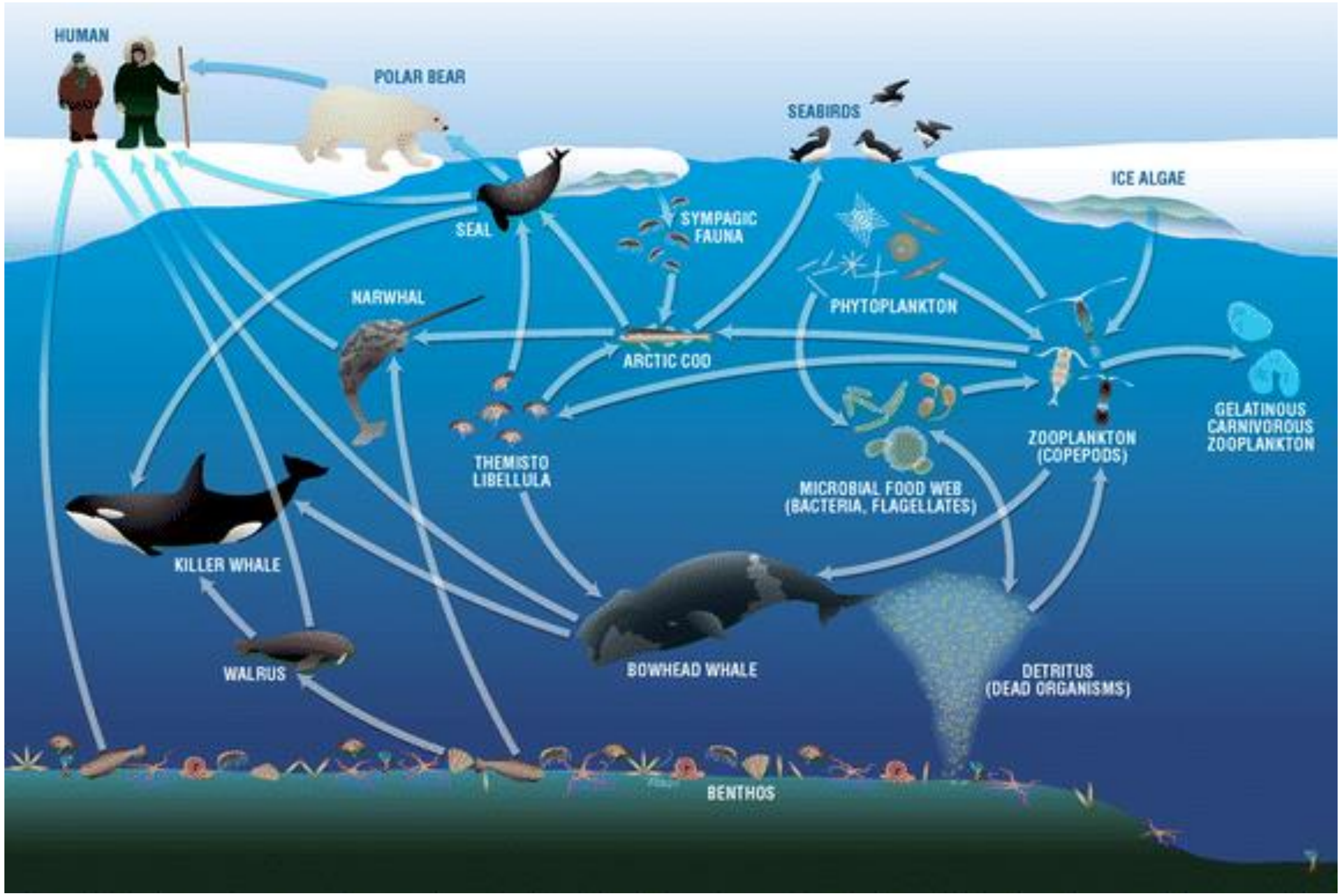
Social Network



Email network

https://www.researchgate.net/figure/Core-Enron-email-network-composed-of-147-nodes_fig1_226840810

Food Web



Edges point from food source to feeder

https://askabiologist.asu.edu/plosable/marine-food-web-collapse

Types of Networks

- What are some examples of networks that you can think of?
- What are some examples of ways in which networks could be used?

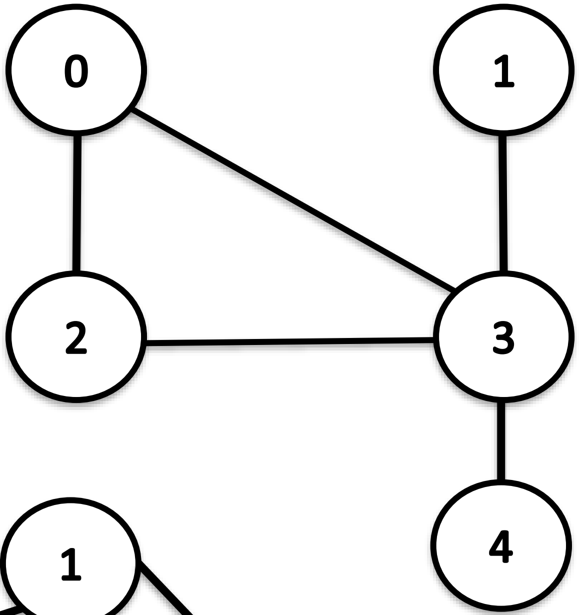
Types of Networks

- Social networks
  - Friendships, family relationships, communications (email, phone), professional groups, physical relationships (e.g., disease spread) etc.
- Information networks
  - Citations among articles, world-wide-web (distinct from the physical communication network), preference networks (e.g., NetFlix)
- Biological networks
  - Metabolic pathways, genetic regulatory networks, neural networks, food web (predator-prey relationships)
- Technological networks
  - Electric power grid, the Internet (physical computer communication network), transportation networks (airline routes, road networks, etc.), electronic circuits, …

Graph Analysis Applications

- Routes in transportation networks
- Spread of diseases/trends/opinions
- Information propagation in the Internet (e.g., viral videos)
- Security (e.g., relationships among people)
- Cascading failures in the electric power grid
- Robustness of telecommunication networks
- Social dynamics (e.g., friendship, authorship, mentorship)
- Many others…

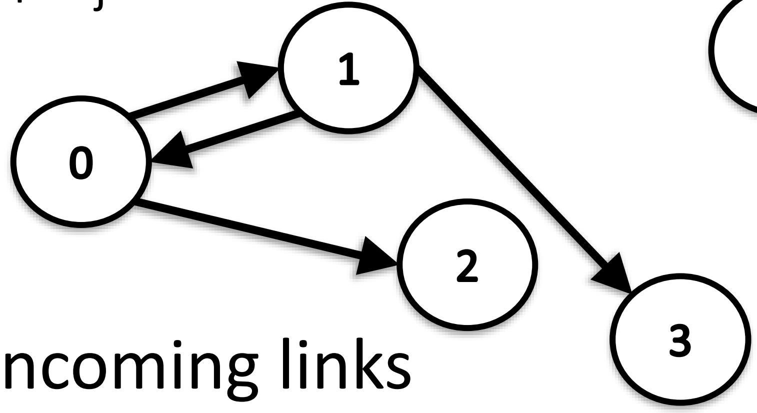
Definitions



- Graph = (V, E)

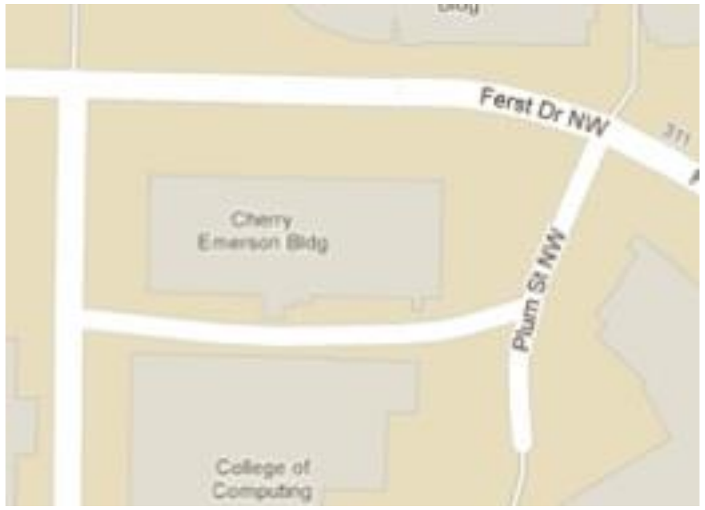
  V = Vertex (node) set = {0, 1, 2, … n-1}

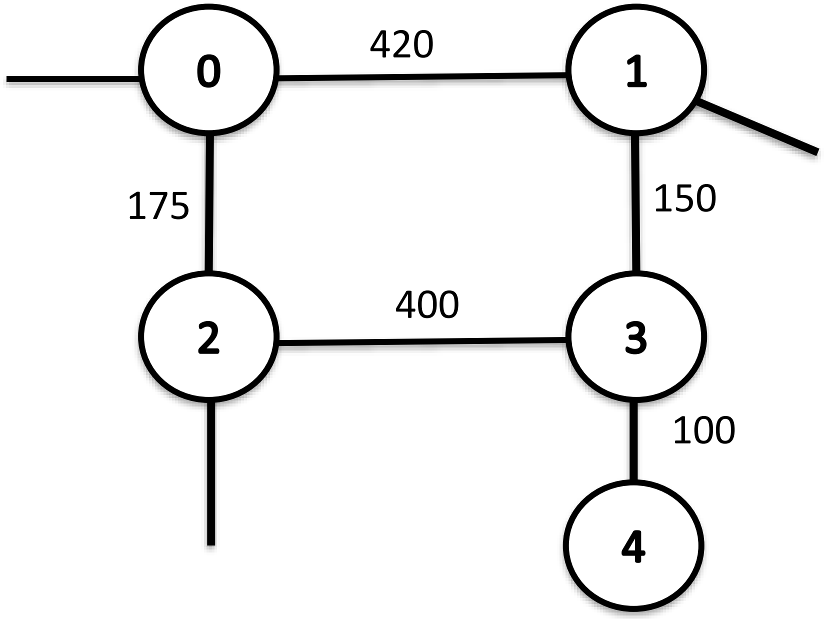
  E = Edge (link) set = $\{e_1, e_2, \ldots e_m\}$ where $e_k = (v_i, v_j)$ where v_i and v_j are vertices in V



- Directed vs. undirected graphs
- Degree(i): number of edges on node i
  - In-degree (directed networks): number of incoming links
  - Out-degree (directed networks): the number of outgoing links
- Often associate properties with vertices and/or edges (e.g., distance)

An Example





- Represent road network as a graph
- Vertices (nodes): intersections
- Edges: road segments
- Weight each edge with a distance

Outline

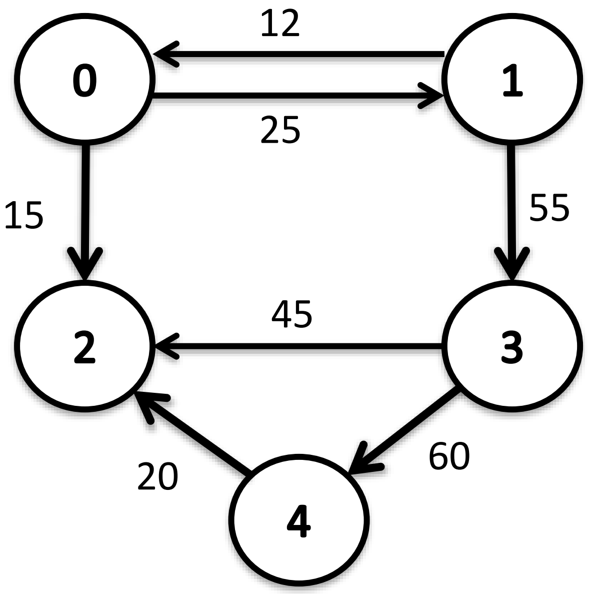
- Applications
- Definitions
- Matrix Representation
  - Directed Graphs
  - Undirected Graphs
  - Pros and Cons
- Linked List Representation
  - Implementation

Representation: Adjacency Matrix

- Label nodes 0, 1, 2, … N-1
- Define matrix A[i, j]: (i, j, = 0, 1, … N-1)

  = 1 if there is a link from node i to node j

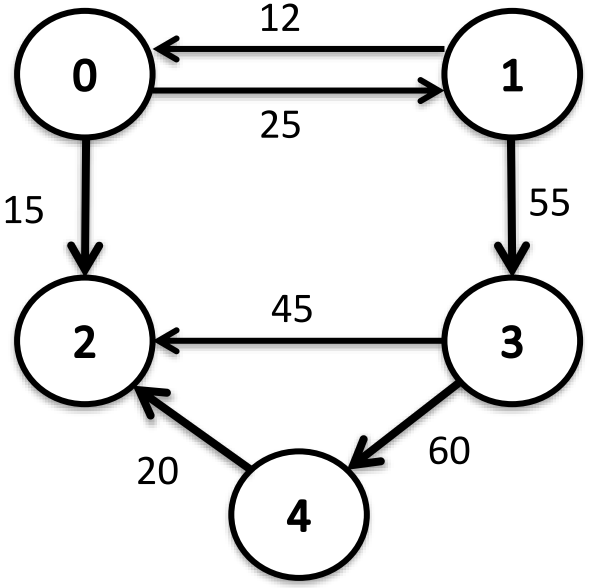
  = 0 if no link exists from i to j
- Alternatively, A[i,j] could represent a quantity such as the distance (or cost, or time) from node i to node j



The blanks actually represent 0s

	0	1	2	3	4
0	-	25	15	-	-
1	12	-	-	55	-
2	-	-	-	-	-
3	-	-	45	-	60
4	-	-	20	-	-

Adjacency Matrix: Observations

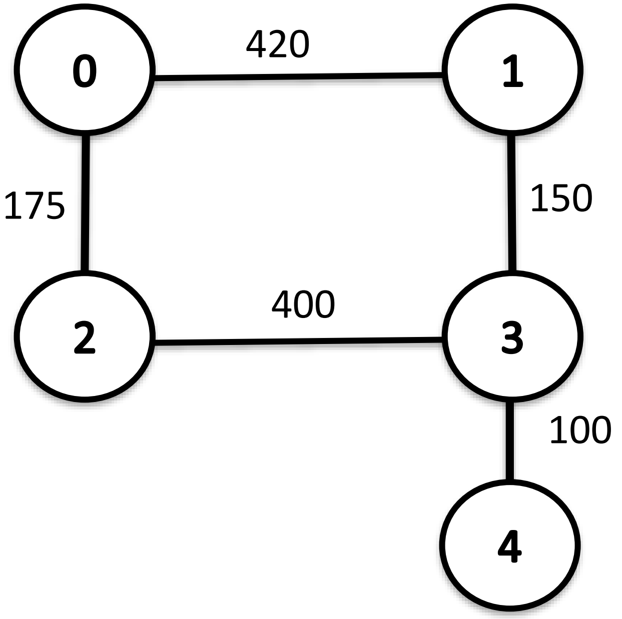


$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & - & 25 & 15 & - & - \\ 1 & 12 & - & - & 55 & - \\ 2 & - & - & - & - & - \\ 3 & - & - & 45 & - & 60 \\ 4 & - & - & 20 & - & - \end{array}$$

- Row i indicates the outgoing links for node i
- Column j indicates the incoming links for node j
- Here the entry indicates the length associated with that link/edge
- Length of a 2-hop path from i to j via node k can be computed as A[i,k] + A[k,j] where i, j, and k are distinct nodes and such a path exists
- Example: Length of path from 0 to 3 via node 1 is A[0,1]+A[1,3] = 25+55 = 80

Adjacency Matrix: Undirected Graphs

- Special case of directed graphs
- For undirected graphs, each link represented by two edges, one in each direction



$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & - & 420 & 175 & - & - \\ 1 & 420 & - & - & 150 & - \\ 2 & 175 & - & - & 400 & - \\ 3 & - & 150 & 400 & - & 100 \\ 4 & - & - & - & 100 & - \end{array}$$

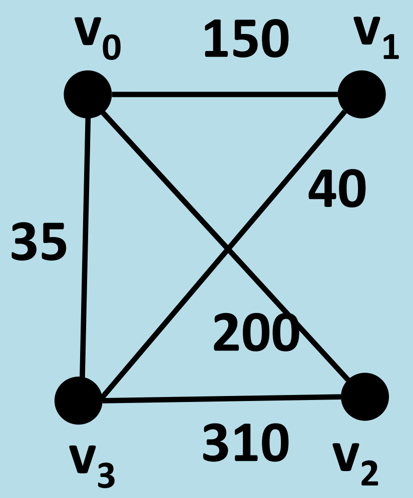
- For undirected graphs, A is symmetric: $A = A^T$

Examples

- Draw the graph given by the adjacency matrix

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

- Use an adjacency matrix to represent the graph.

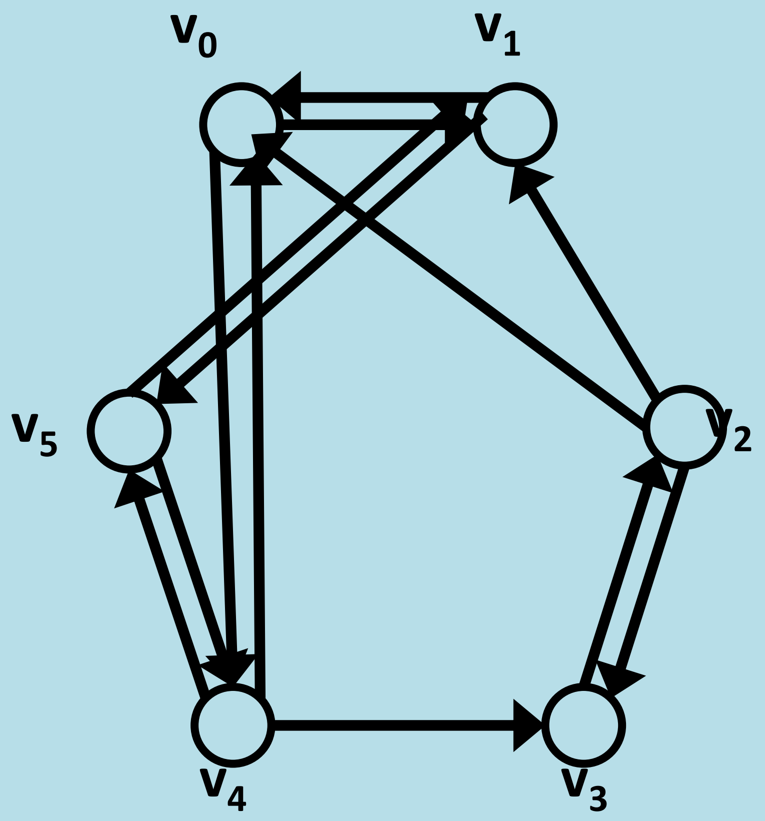


- Is the adjacency matrix of a graph unique? Why or why not?

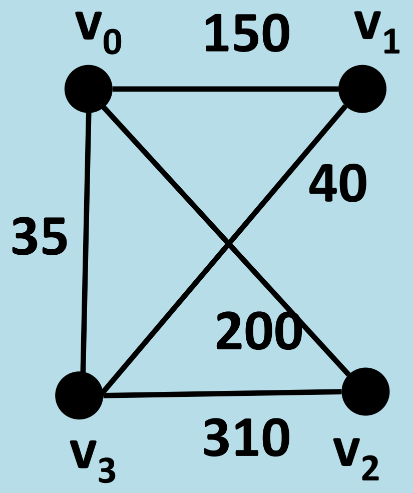
Examples

- Draw the graph given by the adjacency matrix

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$



Examples



- Use an adjacency matrix to represent the graph.

$$A = \begin{bmatrix} 0 & 150 & 200 & 35 \\ 150 & 0 & 0 & 40 \\ 200 & 0 & 0 & 310 \\ 35 & 40 & 310 & 0 \end{bmatrix}$$

- Is the adjacency matrix of a graph unique? Why or why not?
  Once a vertex ordering has been specified, it is. But if we reorder the vertices, the matrix may be different. For example, if row 1 corresponded to v2 and row 2 to v1, the matrix would be different.

Adjacency Matrix Pros & Cons

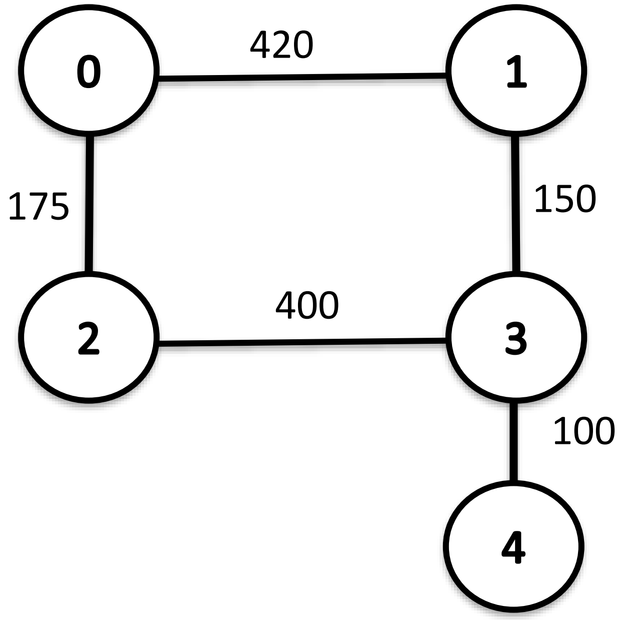
- Easy to determine quickly if there is a link between nodes i and j
- Space = N^2 where N = number of nodes
- Inefficient if graphs are sparse (usual case for large graphs)
  - A graph with 1000 nodes will contain 1,000,000 array elements
  - Most of these entries will be zero or “empty” for most graphs that arise in practice (although there are ways to improve efficiency for such cases)
- For undirected graphs, information is duplicated

Outline

- Applications
- Definitions
- Matrix Representation
  - Directed Graphs
  - Undirected Graphs
  - Pros and Cons
- Linked List Representation
  - Implementation

Adjacency List

- Need a list of vertices; could be stored in a one-dimensional array
- For each vertex i
  - Adj[i] is a list of vertices k where (i, k) ∈ E (neighbors of i) (+ edge weights)
  - Order of nodes in adjacency list may/may not be important

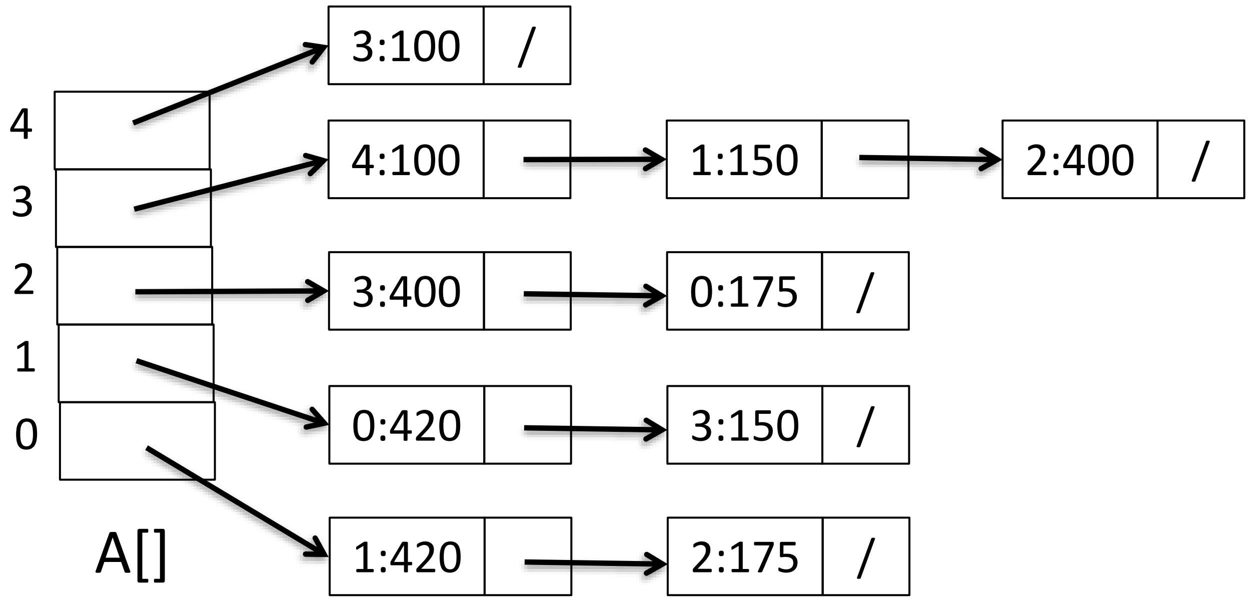


4:	3:100		
3:	4:100	1:150	2:400
2:	3:400	0:175	
1:	0:420	3:150	
0:	1:420	2:175	

- Space?
  - Amount of space is 2*E + N (E edges and N vertices)
  - Preferred representation for sparse graphs
  - Some effort to determine if there is a link between two vertices

Linked List Implementation

- Store each list of edges as a linked list
- A[i] is a pointer to the list of edges to which node i connects

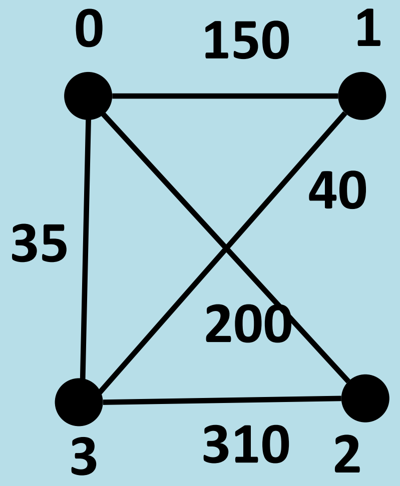


- Implementation issues/choices
  - Sorted vs. unsorted list
  - Operations: insert, delete, find
  - Dynamic graphs that change in size (nodes, edges)

Examples

4:	2:20	
3:	4:60	2:45
2:	NIL	
1:	0:12	3:55
0:	1:25	2:15

- Draw the graph given by the adjacency list.



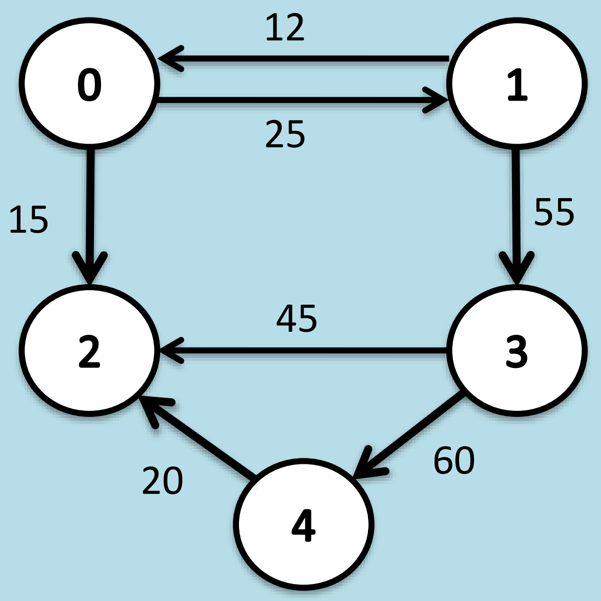
- Use an adjacency list to represent the graph.

- How can edge weights be represented with adjacency matrices/adjacency lists? How about vertex weights?
- What are some other ways you can think of to represent graphs?

Examples

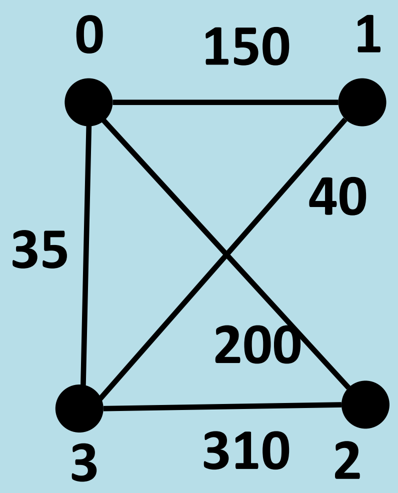
- Draw the graph given by the adjacency list.

4:	2:20	
3:	4:60	2:45
2:	NIL	
1:	0:12	3:55
0:	1:25	2:15



Examples

- Use an adjacency list to represent the graph.



3:	0:35	2:310	1:40
2:	0:200	3:310	
1:	0:150	3:40	
0:	3:35	1:150	2:200

Examples

- How can edge weights be represented with adjacency matrices/adjacency lists? How about vertex weights?
  - Edge weights: With an adjacency matrix, edge weights can be the matrix entries. With an adjacency list, edge weights can be stored with each neighbor node.
  - Vertex weights: In either case, vertex weights may be best stored with the list of vertices. This is easy for an adjacency list in particular. Note that vertex weights cannot be included explicitly in an adjacency matrix.
- What are some other ways you can think of to represent graphs?
  - One idea: build from sparse matrix representations by using a list of ordered pairs to represent edges (works for directed or undirected).
  - Another option: an incidence matrix, with each vertex represented by a row and each edge represented by a column, with nonzero entries for the two vertices associated with each edge in that edge’s column.

Summary

- Graphs arise in many application areas
- Two common ways to represent graphs
  - Adjacency matrix
  - Adjacency list (a type of sparse matrix)
  - Both can be used to represent directed graphs
- Adjacency matrix
  - Space: N^2 elements for N vertices
  - Easy to check if a link exists between two vertices
- Adjacency list
  - More common representation: most large real-world graphs are sparse
  - Space: Number of edges [2*(number of edges) if undirected] + number of vertices
  - Linked list implementation is typically used