Real Lecture 22:
More Clusters more Graphs

Parenthesis on Clusters

Code to cluster all the data can be found here:

- http://bit.ly/10nQ54Q
- The code requires the code for clustering (http://bit.ly/15cApcE) and for reading the data (http://bit.ly/ZCwSQg) to both be in the same folder

Resulting clusters can be found here:

- http://bit.ly/10F0V5A
- Feel free to play with the code and experiment with the effect of different distance metrics

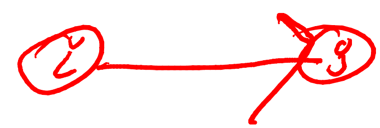
What is a graph

A set of Vertices together with a set of Edges connecting those vertices (V, E)

- The set of vertices is an arbitrary set
  - for this class we assume it is finite

Edges are different depending on the type of graph

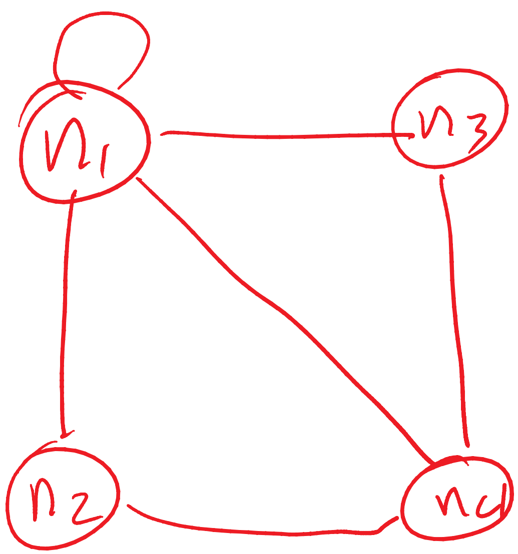
- Undirected: Edge is a pair (v_i, v_j).
  - The pair (v_i, v_j) is equivalent to (v_j, v_i)



- Directed: Edge is a pair (v_i, v_j).
  - The pair (v_i, v_j) is NOT equivalent to (v_j, v_i)
- Weighted: Edge is a triple (v_i, v_j, w).
  - Can be directed or undirected

Example 1: Undirected

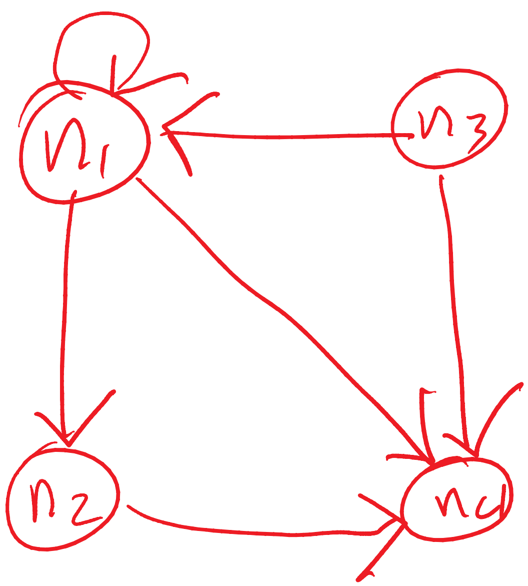
- Example 1: Undirected
  - Vertices $\{n_1, n_2, n_3, n_4\}$
  - Edges $\{(n_1, n_2), (n_3, n_1), (n_1, n_4), (n_2, n_4), (n_1, n_1), (n_3, n_4)\}$



All nodes are reachable from each other

Example 2: Directed

- Example 2: Directed
  - Vertices $\{n_1, n_2, n_3, n_4\}$
  - Edges $\{(n_1, n_2), (n_3, n_1), (n_1, n_4), (n_2, n_4), (n_1, n_1), (n_3, n_4)\}$

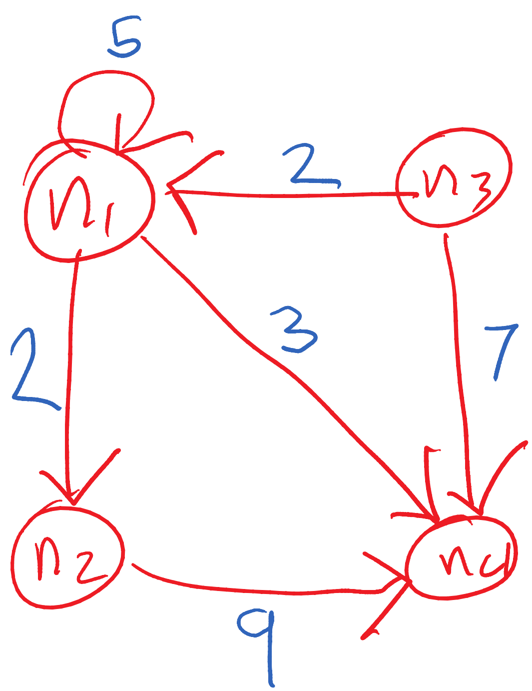


There is no path from n4 to n1 for example

Shortest path from n3 to n4 takes one edge

Example 3: Weighted Directed

- Example 3: Weighted Directed
  - Vertices $\{n_1, n_2, n_3, n_4\}$
  - Edges $\{(n_1, n_2, 2), (n_3, n_1, 2), (n_1, n_4, 3), (n_2, n_4, 9\ ), (n_1, n_1, 5), (n_3, n_4, 7)\}$

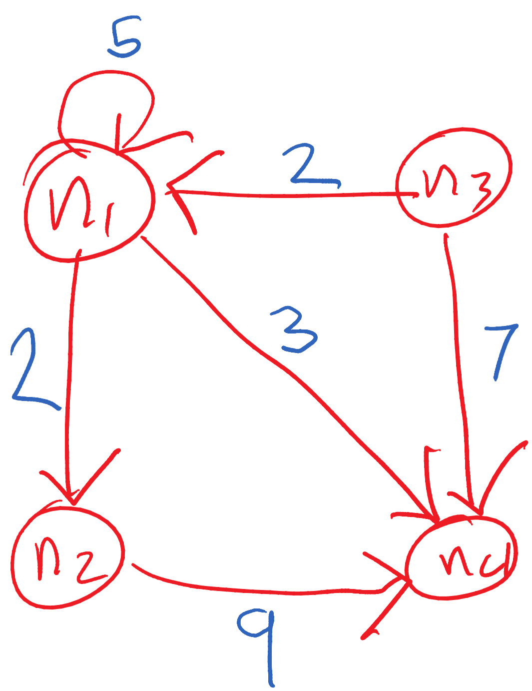


Shortest path from n3 to n4 is not the one-edge path, but the path that goes through n1

Shortest path between any two nodes will never go through a loop

Example 2: Weighted Directed

- Example 2: Weighted Directed
  - Vertices $\{n_1, n_2, n_3, n_4\}$
  - Edges
  $\{(n_1, n_2, 2), (n_3, n_1, 2), (n_1, n_4, 3), (n_2, n_4, 9), (n_1, n_1, \mathbf{-1}), (n_3, n_4, 7)\}$



Shortest path is now undefined for many nodes (it is $-\infty$)
For any path that goes through n1, you can get a shorter path by going around n1 one more time.

Representing Graphs

Representation 1: Set based representation

- Many important operations are expensive

Representation 2: Object oriented

- Node object contains list of neighboring nodes
- List of neighbors contains references pointing to nodes

Representation 3: Matrix

- Very inefficient for sparse matrices

Representation 4: Implicit

- Graph represented as a function
- Sometimes the only mechanism that works when the graph is too big for any other representation

dot: Graph Visualization Language

Very simple syntax:

```
digraph name{
    v1 -> v2[label="something"];
    v2->v3;
}
```

You can find many visualizers on the web

- GraphViz is probably the most popular one
  - http://www.graphviz.org/
  - There is a simple web viewer here: http://sandbox.kidstrythisathome.com/erdos/

    but it only supports a small number of nodes and edges if you want to play with this, it's better to download the stand alone version

Key algorithmic questions

Is there a path between two nodes?

What is the shortest path between two nodes?

Can we order the nodes in a graph so all nodes come before their successors in the graph?