Graphs

Fall Semester

Topic Outline

- Definitions
  - Node
  - Edge
  - Path
- Examples
- Representation
  - Edge list
  - Neighbor list
  - Adjacency matrix
- Python Code
  - Hash table
  - Key is node
  - Value is list of neighbors
- Connected Component
  - Metrics
  - Timing
  - Random Ladders
- Up Next: Search

Definitions (1)

- Node. Also called a vertex.
- Edge. A connection between two nodes. An edge may have a numeric weight. An edge may be directed.
- Graph. A collection of nodes and edges.
- Path. A sequence of nodes; for each non-terminal node an edge must exist connecting it to the next node in the sequence.
- Cycle. A path whose starting and ending nodes are the same.
- Tree. A graph with no cycles.
- Network. A weighted, directed graph.

Definitions (2)

- Loop. An edge connecting a node to itself.
- Degree. The total number of times a node appears in any edge.
- Hamiltonian path. Contains each *node* exactly once.
- Eulerian cycle. Contains each *edge* exactly once.
- Minimal spanning tree. A subgraph that is a tree, contains all the nodes, and has a total edge cost that is as small as possible.
- Connected component. A subgraph that contains every node connected by a path to any other node in the component.
- Diameter. The number of edges in the longest shortest path of a connected component.

Example Graph: Seven Bridges of Königsberg (Euler, 1736)

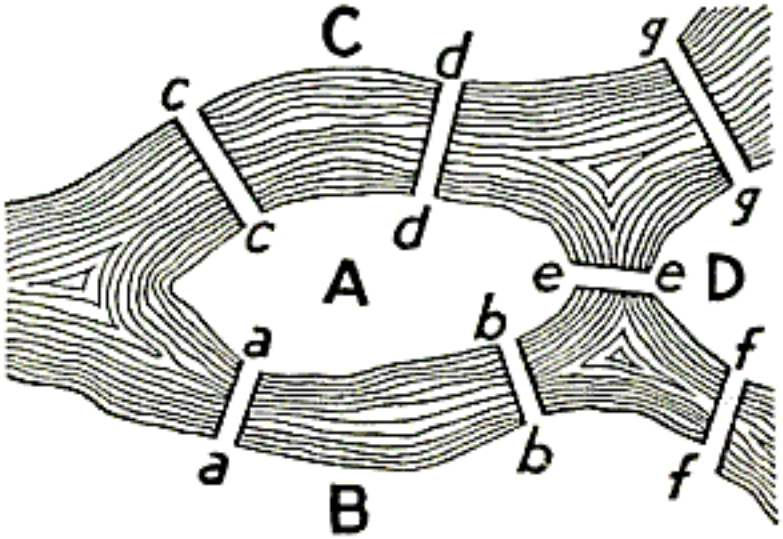


FIGURE 98. *Geographic Map: The Königsberg Bridges.*

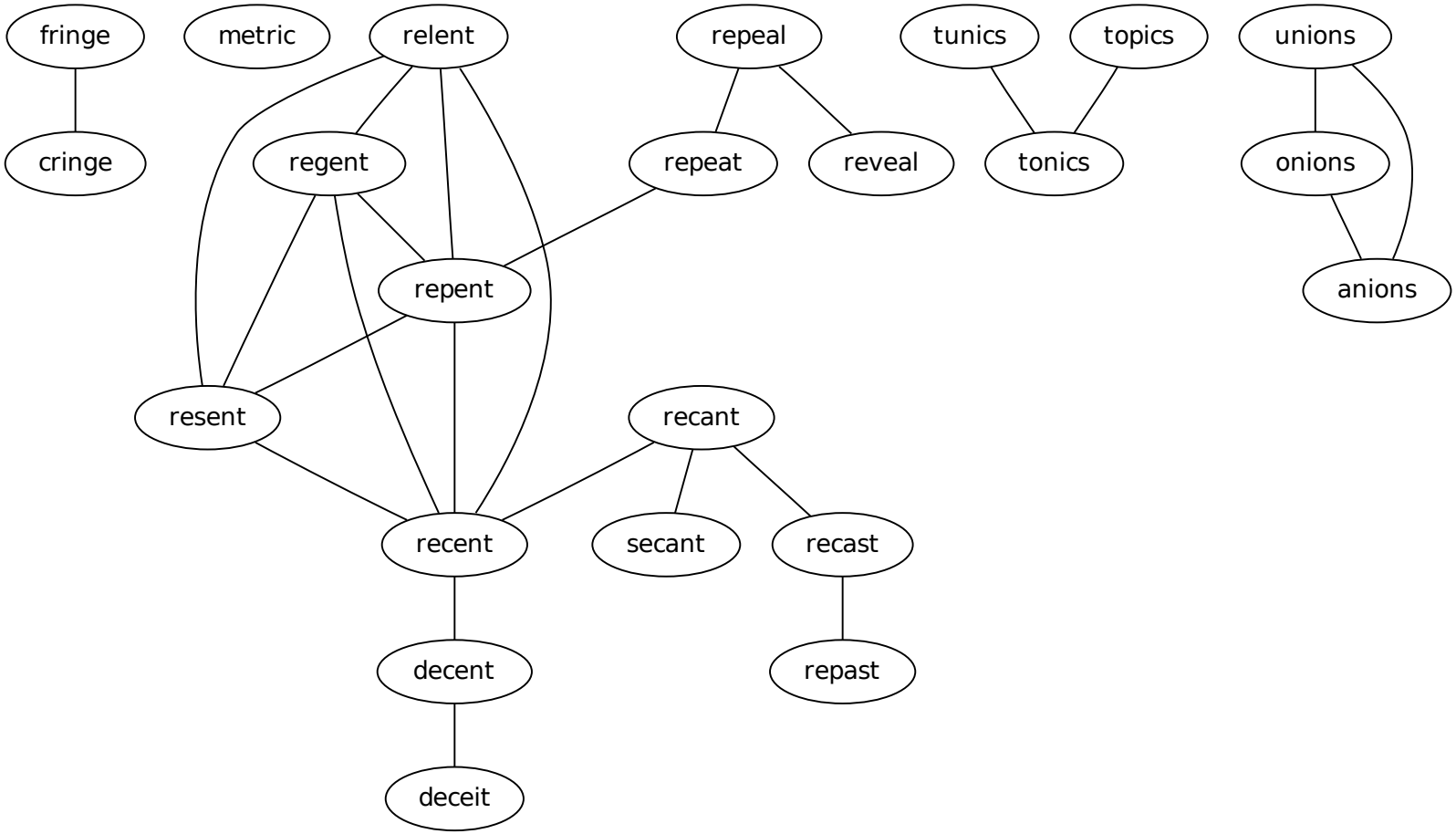
Weisstein, Eric W. "Königsberg Bridge Problem." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/KoenigsbergBridgeProblem.html

Example Graph: Travelling Salesman Problem

- Nodes are cities that must be visited.
- Connections between cities are weighted, undirected edges.
- The goal is to find an optimal Hamiltonian cycle.
- NP-Complete
  - A given solution can be *verified* in polynomial time.
  - If this particular problem can be *solved* quickly then so can every other NP problem.
  - But, combinatorial explosion (i.e., exponential growth).
- Clay Institute Millenium Prize Problem.

Example Graph: Word Ladders

- The words are the nodes. We will consider six-letter words only.
- An unweighted, undirected edge exists between any two nodes that differ by exactly one letter.
- A puzzle consists of the starting and ending words. Its solution (the word ladder) is any path connecting those two words.
- Invented by Lewis Carroll on March 29, 1879, in *Vanity Fair*.
- Alternatives include adding or removing a letter at each step, or substituting an anagram for the current word.
- See also, Gray codes.


fringe
metric
relent
repeal
tunics
topics
unions
cringe
regent
repeat
reveal
tonics
onions
repent
anions
resent
recant
recent
secant
recast
decent
repast
deceit

Other Examples

- Six degrees of Kevin Bacon.
  - See oracleofbacon.org, an old UVA website.
- Transportation networks: ground, water, air.
- Websites linked together form a directed graph.
- Unstructured mesh for computational physics on a complicated geometry using triangles or tetrahedrons.
- Constraint satisfaction such as sudoku or map coloring.
- A finite state machine as an edge-labeled directed graph.

Representation

- Edge list. A list containing pairs of connected nodes. Data files may take this form (e.g., a list of states bordering each other).
- Neighbor list. Also called an adjacency list, the sparse matrix analog to the adjacency matrix. This will be our data structure of choice and is easily implemented using a hash table.
- Adjacency matrix. Index each node. Then, element A_{ij} is 0 if there is no edge from node i to node j, otherwise it gives the weight of the edge connecting i to j.

Properties of the Adjacency Matrix

- If the graph is undirected then the matrix is symmetric.
- If the graph is unweighted then all nonzero elements are 1.
- The elements of A^n for an unweighted graph indicate the number of paths of length n between any two nodes. Wow!
- In a complete graph there is an edge connecting every possible pair of the V vertices (nodes), so there are $E = \frac{V(V-1)}{2}$ edges.
- A sparse graph has $E \approx O(V)$ so the A_{ij} are mostly 0.
- A dense graph has $E \approx O\left(V^2\right)$ so the A_{ij} are mostly 1.

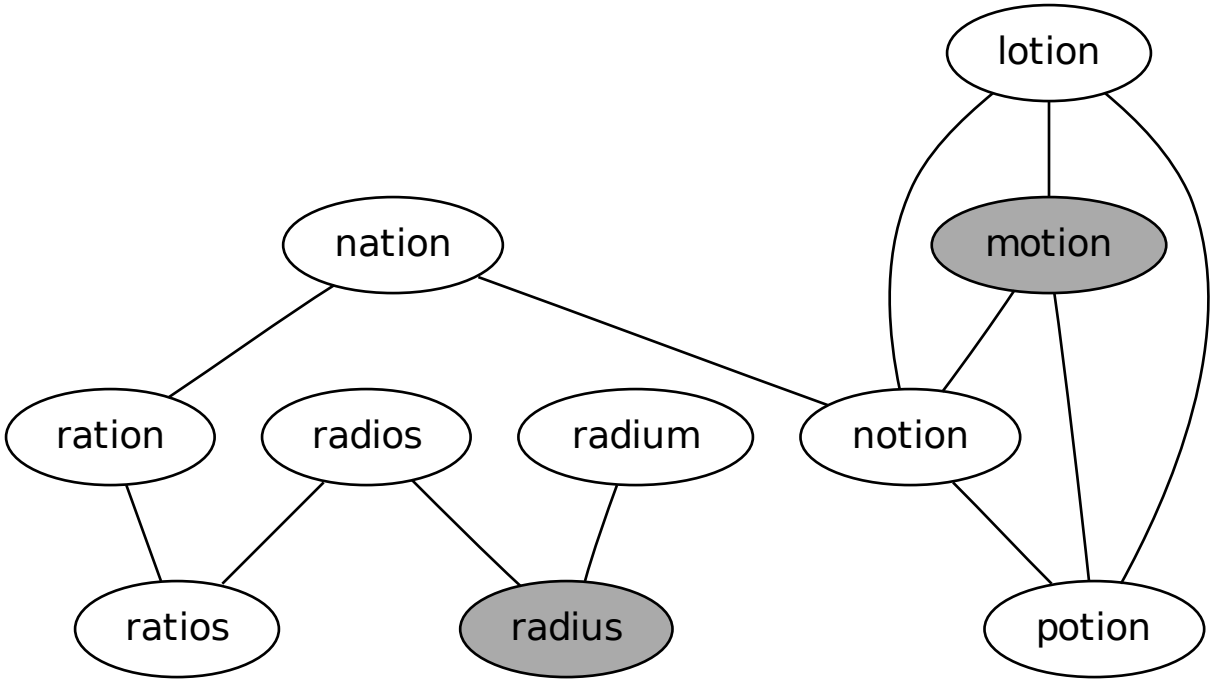
Hash Table

```
ht={}                          # empty "dictionary"
ht['cat']=7                    # (key,value) pairs
ht[0]=[3.14]
ht[5]=['dog',1]
print ht['cat']                # 7
print ht[0]                    # [3.1400000000000001]
print ht[5]                    # ['dog',1]
print ht.keys()                # [0,5,'cat']
```

Example Program: Miscellaneous

- import statement using * (asterisk)
- time module, tic and toc, difference
- convert a string into a character list
- shuffle a list or choice from a list
- list comprehension, range function
- append and pop, use for stack/queue
- difference between randint and random
- file output, explicit newline character

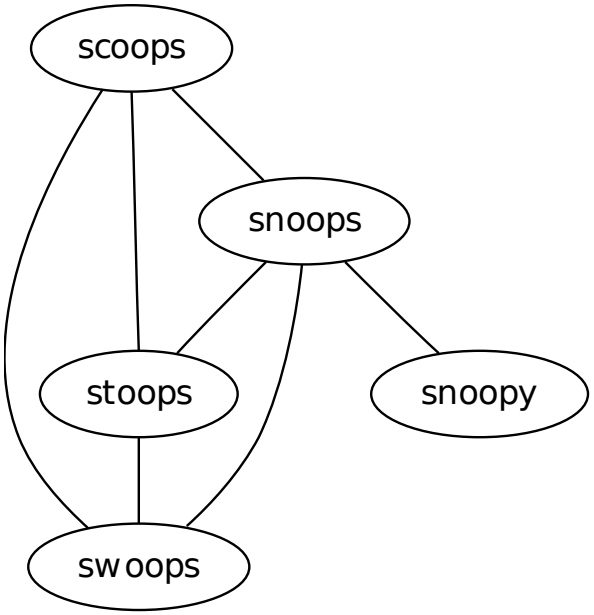
Example Connected Component: FROM radius TO motion



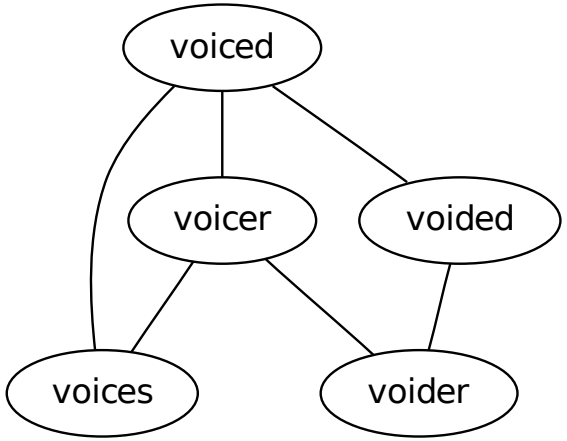
```
KEY     :  VALUE
lotion  :  [motion,notion,potion]
motion  :  [lotion,notion,potion]
nation  :  [notion,ration]
notion  :  [lotion,motion,nation,potion]
potion  :  [lotion,motion,notion]
radios  :  [radius,ratios]
radium  :  [radius]
radius  :  [radios,radium]
ration  :  [nation,ratios]
ratios  :  [radios,ration]
```

More Connected Components (1)

```
Seven edges
Diameter = 2
```

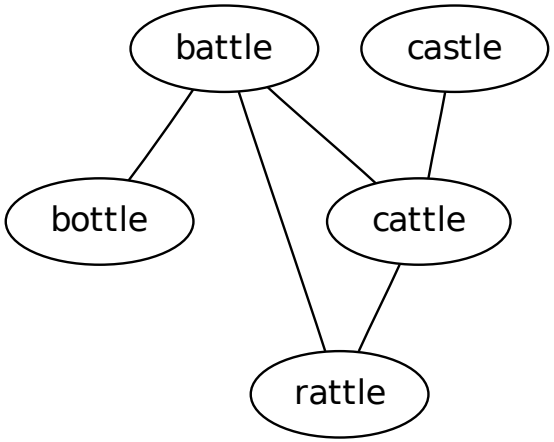



```
Six edges
Diameter = 2
```

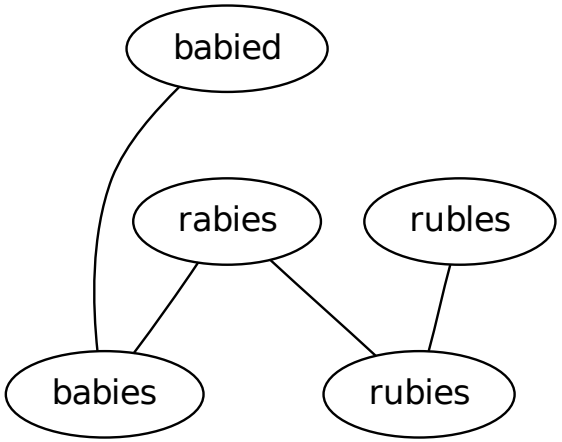


More Connected Components (2)

```
Five edges
Diameter = 3
```




```
Four edges
Diameter = 4
```



Lab Assignment: Random Ladders

- Input the dictionary of six-letter words.
- Construct a random sequence of words such that:
  - The first word has no corresponding letters in common with the last word (e.g., FROM acorns TO stewed).
  - All the words are unique.
  - Consecutive words differ by exactly one letter.
- For testing purposes assume "acorns" is the starting word.
- Output the *entire* sequence of words to the console.
- Output *only* the starting and ending words to a text file.