

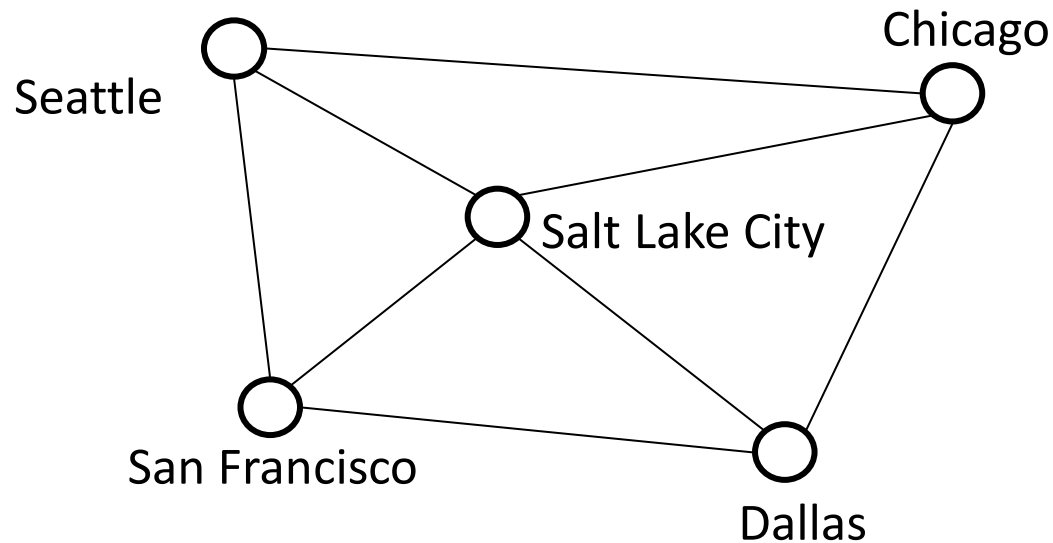
# Graphs

Rob Thompson

UW CSE 160

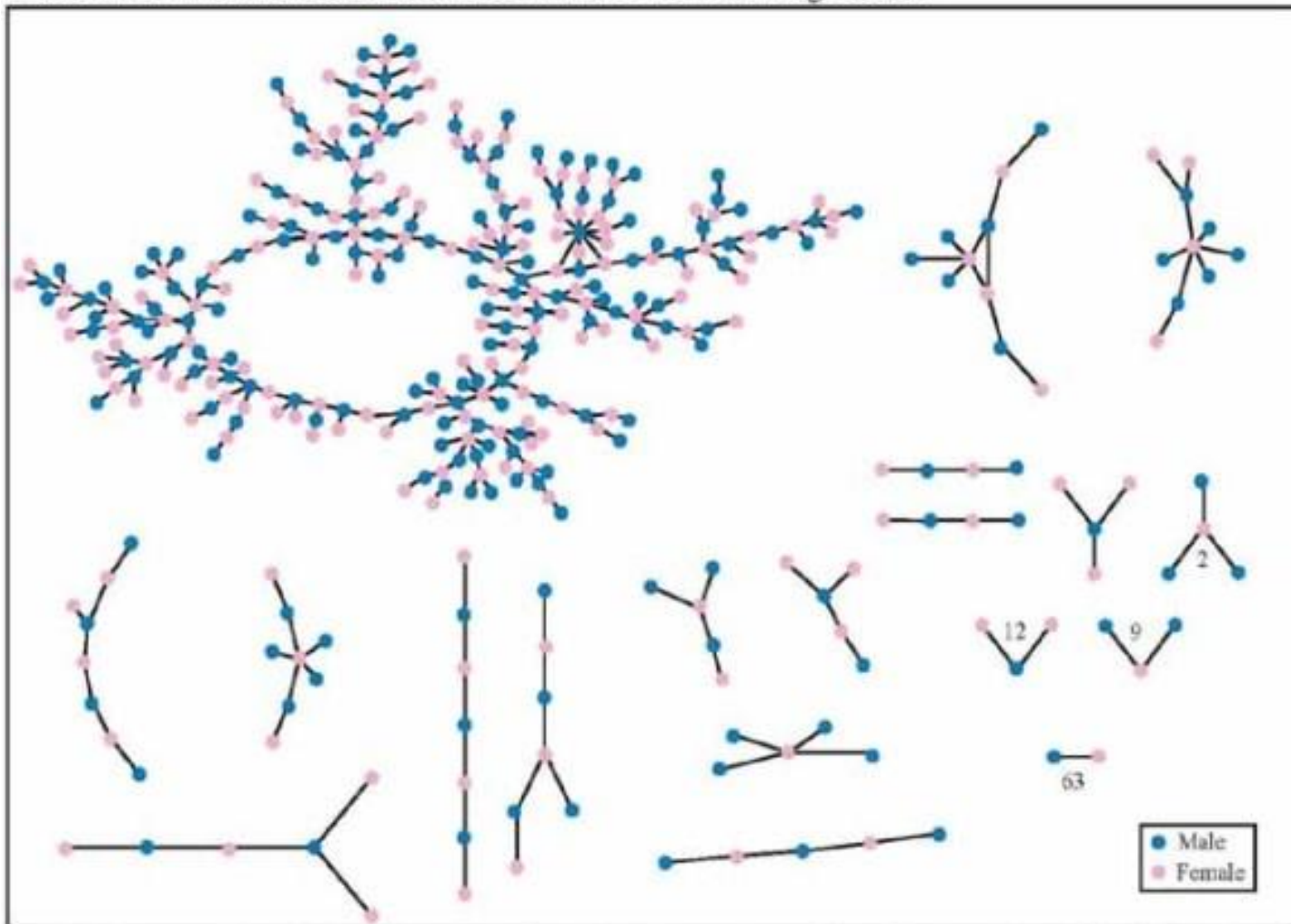
Winter 2021

# A graph contains nodes and edges





## The Structure of Romantic and Sexual Relations at "Jefferson High School"



Each circle represents a student and lines connecting students represent romantic relations occurring within the 6 months preceding the interview. Numbers under the figure count the number of times that pattern was observed (i.e. we found 63 pairs unconnected to anyone else).

+ 350 students in no romantic and/or sexual relationship

From: "Chains of Affection: The Structure of Adolescent Romantic and Sexual Networks", *American Journal of Sociology*, by Peter Bearman of (Columbia), James Moody (Ohio State), and Katherine Stovel (U. of Washington);

# Graphs

- A graph can be thought of as either:
  - a collection of edges
    - Each edge represents some relationship
  - for each node, a collection of neighbors
    - The neighbors are those connected by an edge

# Operations on a graph

Creation:

- Create an empty graph

Querying:

- Look up a node: Does it exist? What are its neighbors?
- Look up an edge (= a pair of nodes): does it exist? (You know the nodes it connects.)
- Iterate through the nodes or edges

Modification:

- Add/remove a node
- Add/remove an edge

# networkx Graph Library

- Used in Homework 5
- Included in the Anaconda Distribution
- <https://networkx.github.io/documentation/stable/tutorial.html>

```
import networkx as nx
g = nx.Graph() #create an empty graph
g.add_node(1) #add one node to graph
g.add_node(2)
g.add_edge(1, 2) #add one edge
print(g.nodes()) #print all nodes
print(g.edges()) #print all edges
```

Note: It is also o.k. to just add an edge before you add the individual nodes; the nodes will be added for you in that case.

```
import networkx as nx
import matplotlib.pyplot as plt

g = nx.Graph()          # Creates a graph

g.add_edge(1, 2)       # Adds edge from node 1 to node 2
g.add_edge(1, 3)
g.add_node(4)          # Adds node 4
print("Edges:", g.edges())
print("Nodes:", g.nodes())
print("Neighbors of node 1:", list(g.neighbors(1)))

assert len(g.nodes()) == 4
assert len(g.edges()) == 2

nx.draw_networkx(g)    # Draw the graph
plt.show()             # Show the graph
```