Graph Theory and Concepts

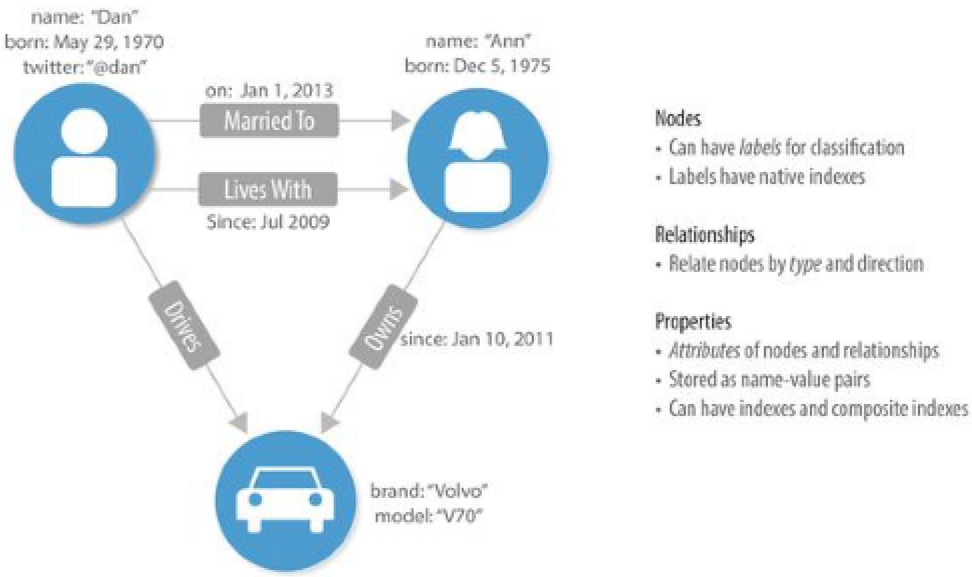
In this lecture

- Framework and terminology for graph algorithms.
- Focus on concepts relevant for practicioners.
- Overview of graph algorithms we will cover.

Labeled Property Graph Model

- It contains nodes and relationships.
- Nodes contain properties (key-value pairs).
- Nodes can be labeled with one or more labels.
- Relationships are named and directed, and always have a start and end node.
- Relationships can also contain properties.

Example



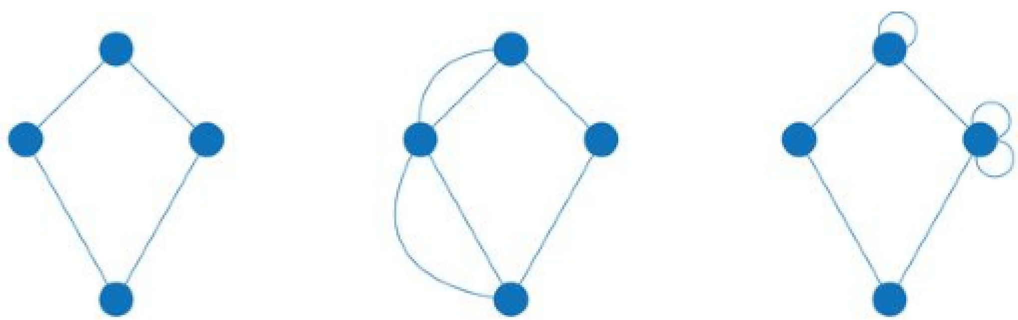
name: "Dan"
born: May 29, 1970
twitter: "@dan"
name: "Ann"
born: Dec 5, 1975
on: Jan 1, 2013
Married To
Lives With
Since: Jul 2009
Drives
Owns
since: Jan 10, 2011
brand: "Volvo"
model: "V70"
Nodes
• Can have *labels* for classification
• Labels have native indexes
Relationships
• Relate nodes by *type* and direction
Properties
• *Attributes* of nodes and relationships
• Stored as name-value pairs
• Can have indexes and composite indexes

Paths and subgraphs

- A **subgraph** is a graph within a larger graph. These can be useful for filters in a focused analysis.
- A **path** is a group of nodes and their connecting relationships.

Graph types

- **Simple:** nodes only have one relationship between them.
- **Multigraph:** multiple relationships allowed.
- **Pseudo graph:** multiple relationships and loops allowed.



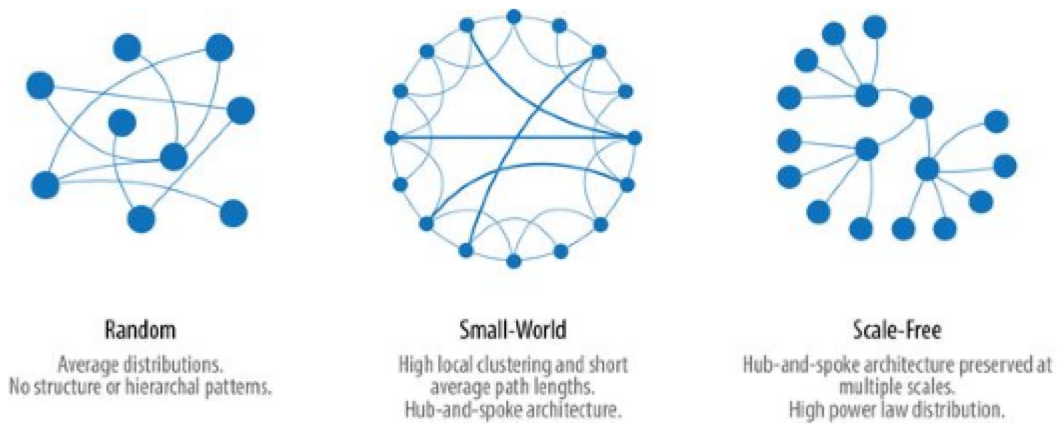
Simple Graph
Node pairs can only have one relationship between them.

Multigraph
Node pairs can have multiple relationships between them.

Graph (also Pseudograph)
Node pairs can have multiple relationships between them. Nodes can loop back to themselves.

Graph structure

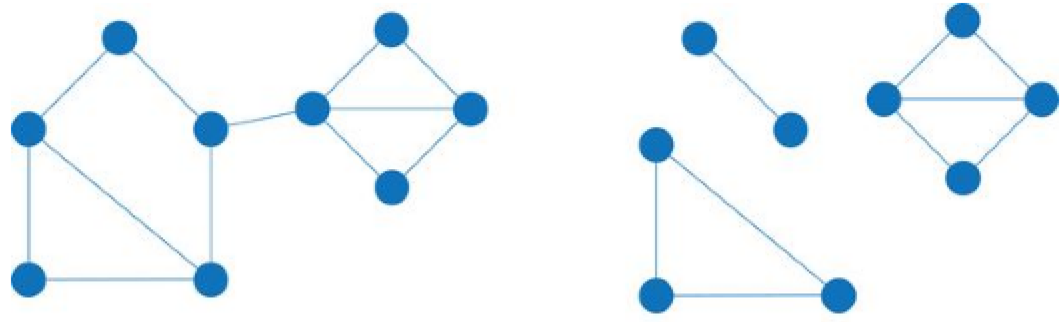
- **Random networks:** unobserved in practice.
- **Small-world networks:** local connections with global reach.
- **Scale-free networks:** self-similarity structure.



Flavors of graphs

Connected vs disconnected

- **Connected:** there is a path between every pair of vertices.
- Disconnected graphs may cause problems for some algorithms.



Connected Graph

Disconnected Graph
Includes 3 components.

Weighted vs unweighted

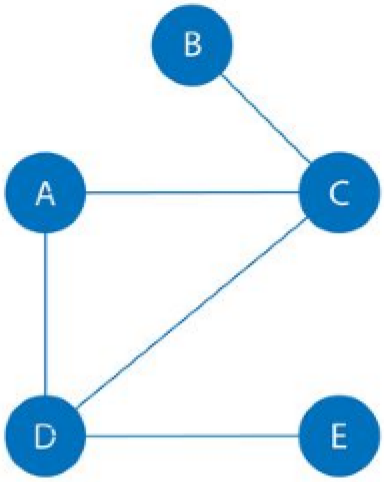
- Sometimes it is useful to quantify the *strength* of a relationship with weights.
- Compare the shortest path between A and E in both cases.

Unweighted

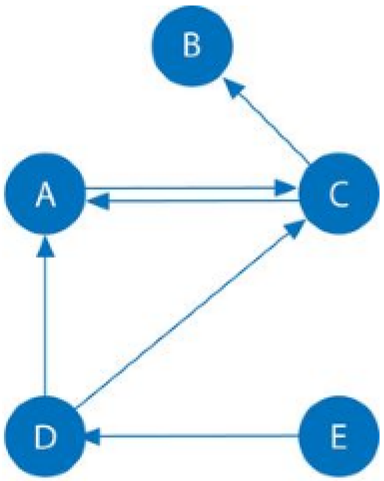
Weighted

Undirected vs directed

- Relationships may not be symmetrical!
- For instance, one-way roads, "likes", etc.



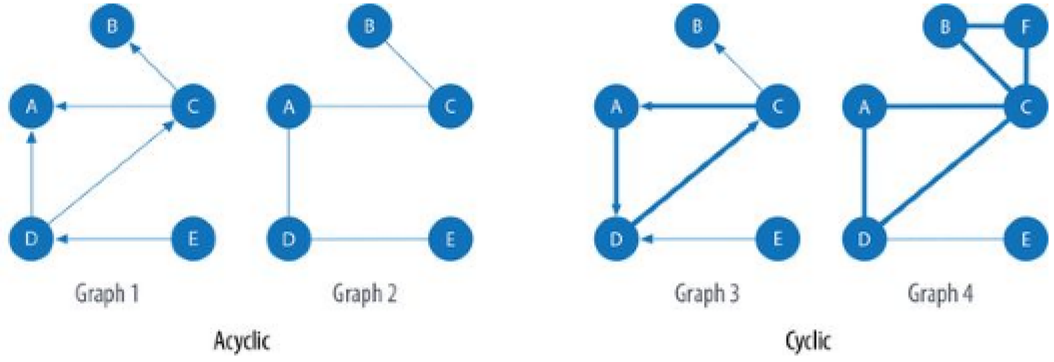
Undirected



Directed

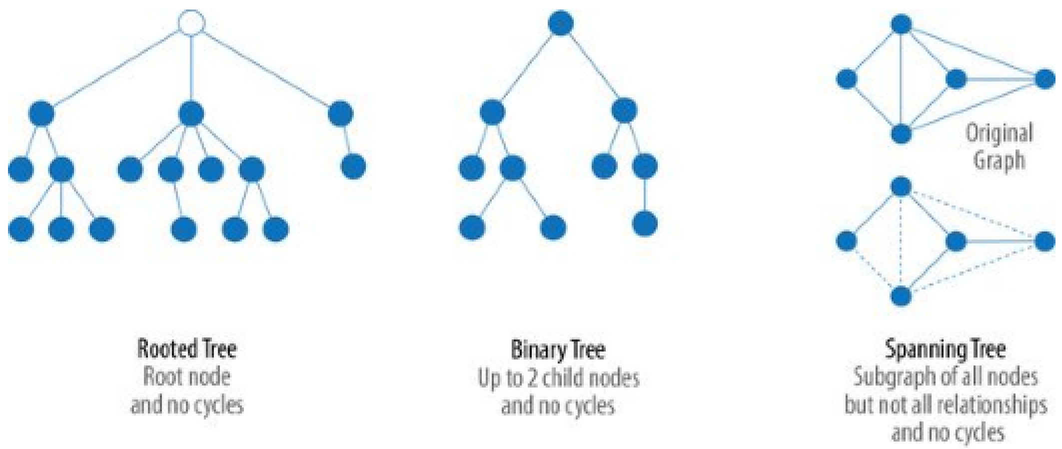
Cyclic vs acyclic

- Cyclic graphs can occur sometimes (groups of friends).
- Acyclic graphs arise in genealogy, version histories, scheduling problems.



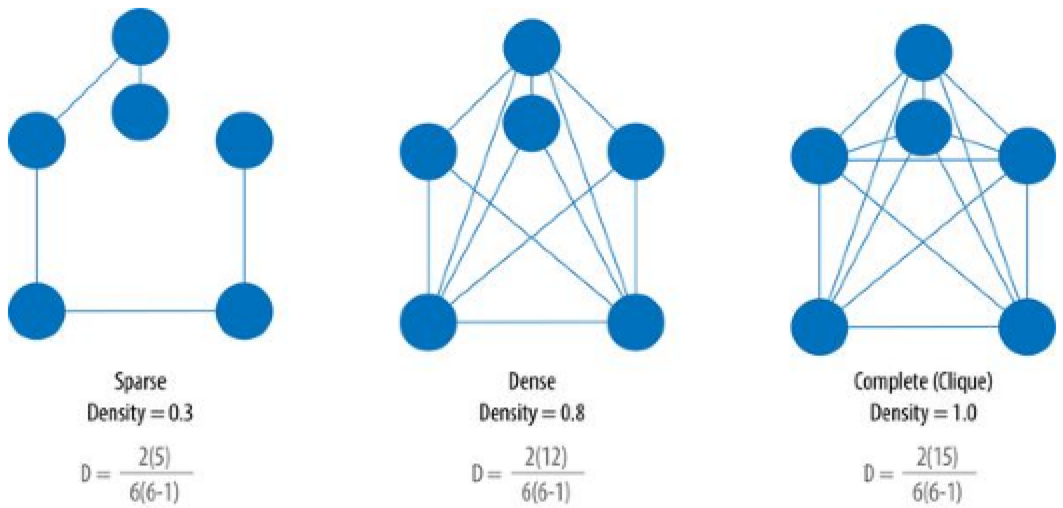
Trees

- Acyclic graph such that any two nodes are connected by exactly one path.
- Key role in network design, data structures, search optimization.



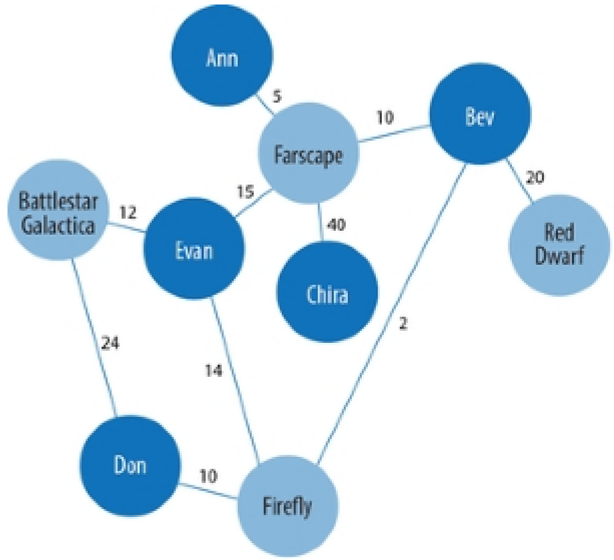
Sparse vs dense

- Max edges on an N vertex graph: $E_{\max} := \frac{N(N-1)}{2}$.
- Density: $\frac{R}{E_{max}} = \frac{2 \cdot R}{N(N-1)}$.
- Extremely sparse or dense graphs can give meaningless results sometimes!



Monopartite, bipartite and projections

- Many networks have multiple node and relationship types.
- If that is the case, graphs can be **bipartite** or **k-partite**, depending on the number of nodes/relationship types.
- However, graph algorithms need only one node type and one relationship type (monopartite graphs).
- Hence some preprocessing is needed, called **projection**.

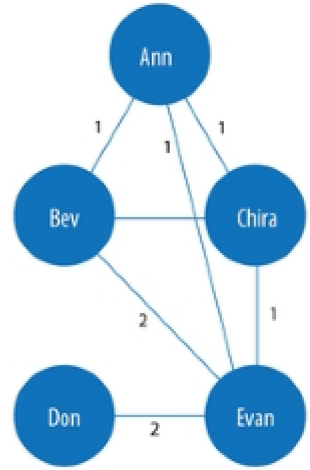


Graph 1

Viewers and TV Shows

Bipartite Graph

Relationship weights = Number of episodes watched

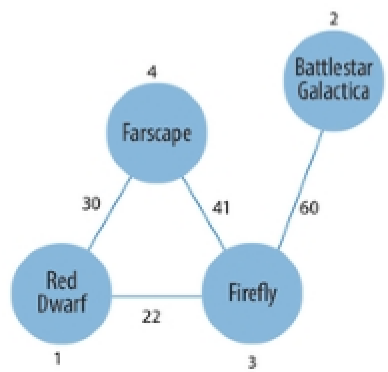


Graph 2

Projection of Viewers

Monopartite Graph

Relationship weights = Number of shows in common



Graph 3

Projection of TV Shows

Monopartite Graph

Node weights =Number of active viewers

Relationship weights = Combined episodes watched by viewers in common

Types of graph algorithms

- **Pathfinding:** Finding shortest paths between graphs.
- **Centrality:** Which nodes are more important in a network?
- **Community detection:** Most real-world networks have substructures or communities of more or less independent subgraphs. For example, *echo chambers* or *filter bubble effects*.