WEB INTELLIGENCE

Lecture 10:
Introduction to Graphs

Russa Biswas

November 18, 2025

AALBORG UNIVERSITY

Based on slides by Jure Leskovec, Jon Crowcroft

Graphs

Image credit: Medium

Social Networks

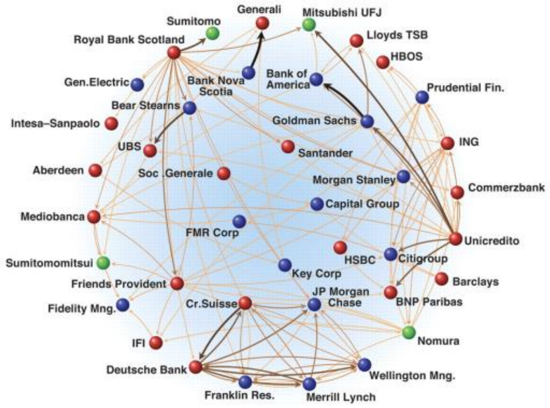
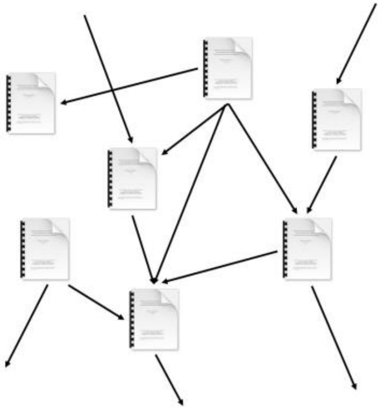


Image credit: Science

Economic Networks

Image credit: Lumen Learning

Communication Networks



Citation Networks

Image credit: Missoula Current News

Internet

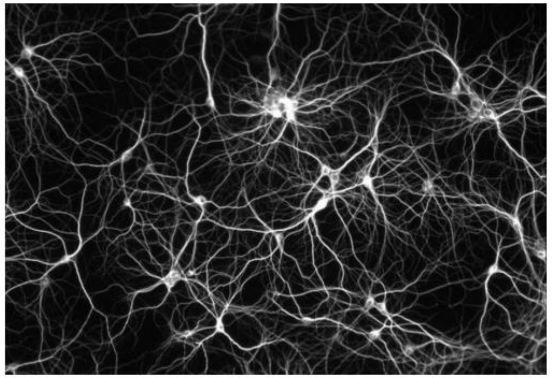
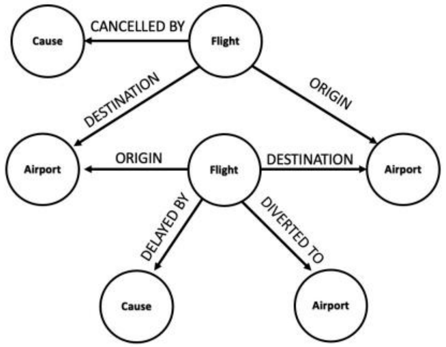


Image credit: The Conversation

Networks of Neurons

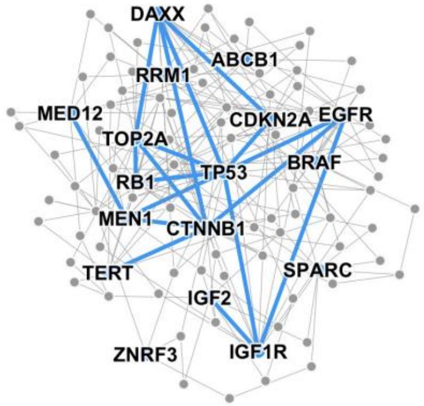
Graphs – More examples



Event Graphs

Image credit: SalientNetworks

Computer Networks



Disease Pathways

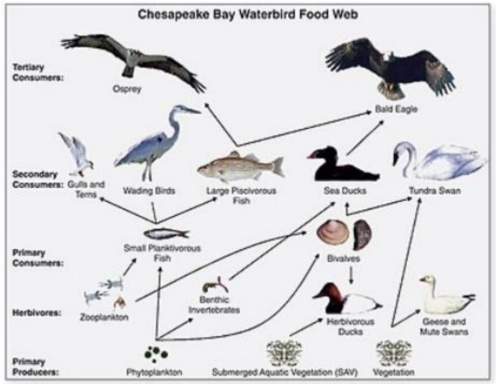


Image credit: Wikipedia

Food Webs

Image credit: Pinterest

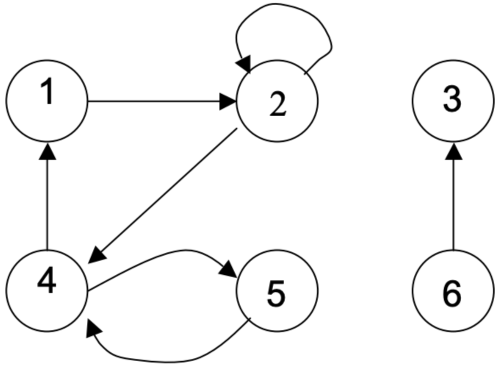
Particle Networks

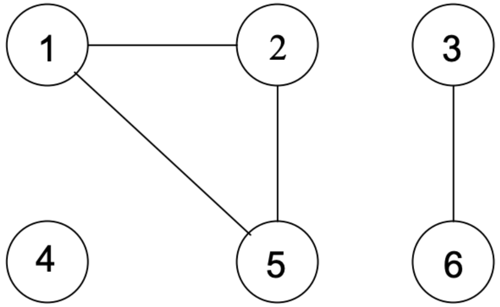
Image credit: visitlondon.com

Underground Networks

Graphs

- **What is a Graph?**
  - Informally a graph is a set of nodes joined by a set of lines or is a set of nodes joined by a set of lines or arrows.
- **Why Graphs?**
  - Representing a problem as a graph can provide a different point of view
  - Representing a problem as a graph can make a problem much simpler
    - More accurately, it can provide the appropriate tools for solving the problem
  - Therefore, Graphs are used for describing and analyzing entities with relations/interactions
- **Network = graph**





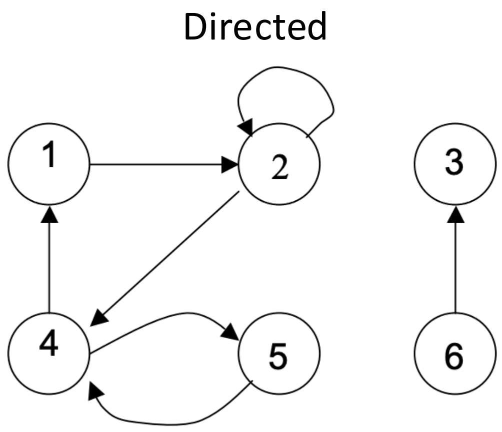
What is Network Theory?

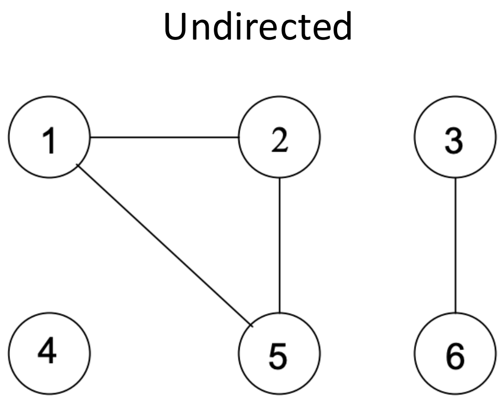
- Network theory provides a set of techniques for analysing graphs
- Complex systems network theory provides techniques for analysing structure in a system of interacting agents, represented as a network
- Applying network theory to a system means using a graph-theoretic representation
- Most complex systems are graph-like

Definitions

Graph

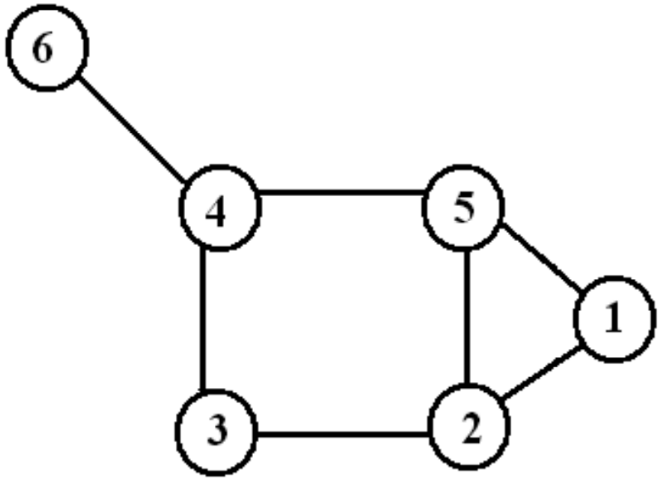
- G is an ordered triple G:=(V, E, f)
  - V is a set of nodes, points, or vertices.
  - E is a set, whose elements are known as edges or lines.
  - f is a function
    - maps each element of E
    - to an unordered pair of vertices in V





Definitions

- Vertex
  - Basic Element
  - Drawn as a node or a dot.
  - Vertex set of G is usually denoted by V(G), or V
- Edge
  - A set of two elements
  - Drawn as a line connecting two vertices, called end vertices, or endpoints.
  - The edge set of G is usually denoted by E(G), or E.

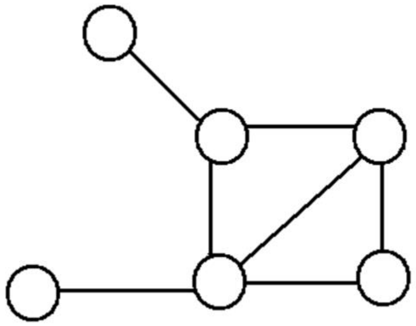


- V:={1,2,3,4,5,6}
- E:={{1,2},{1,5},{2,3},{2,5},{3,4},{4,5},{4,6}}

Definitions

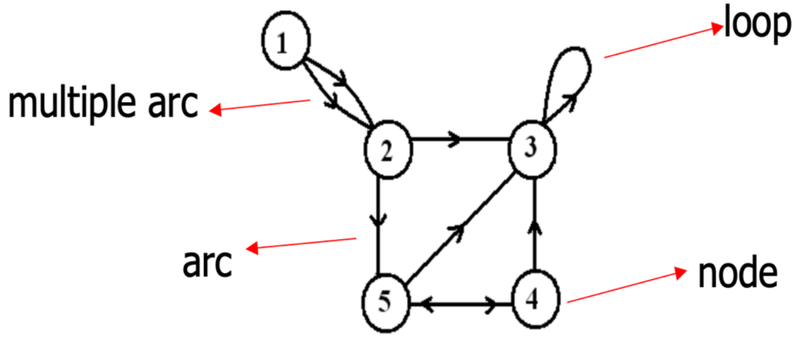
Simple graphs

- Simple graphs are graphs without multiple edges or self-loops.



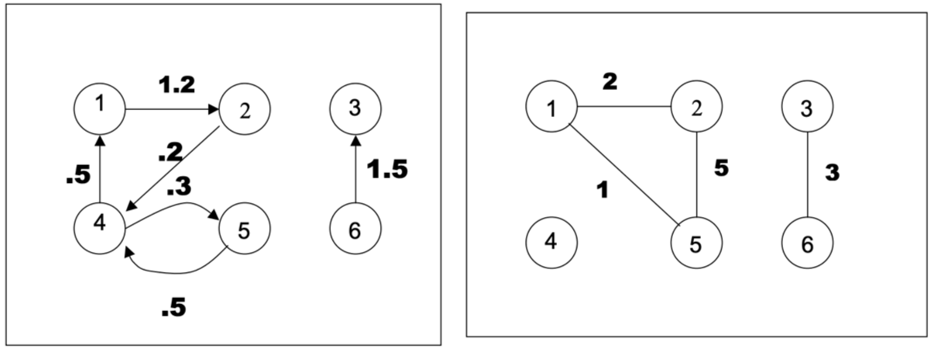
Directed Graphs (Digraph)

- Edges have directions: An edge is an ordered ordered pair of nodes



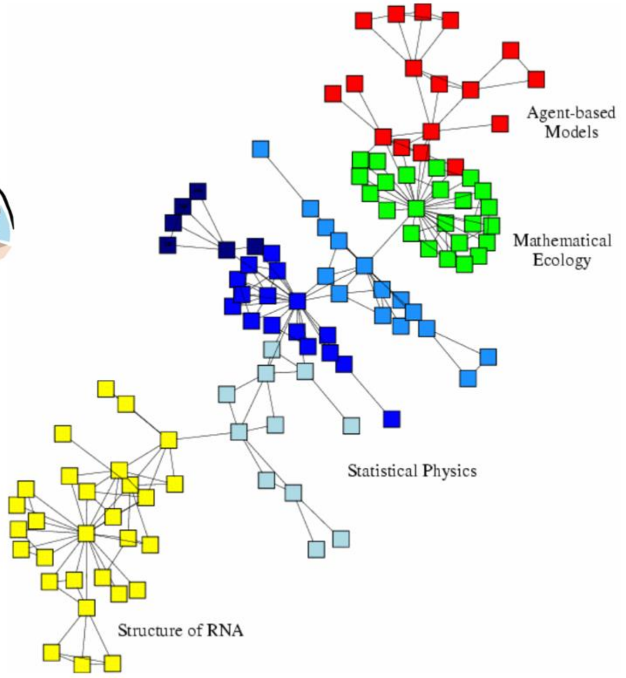
Weighted Graphs

- is a graph for which each edge has an associated weight, usually given by a weight function w: E → R.



Structures and structural metrics

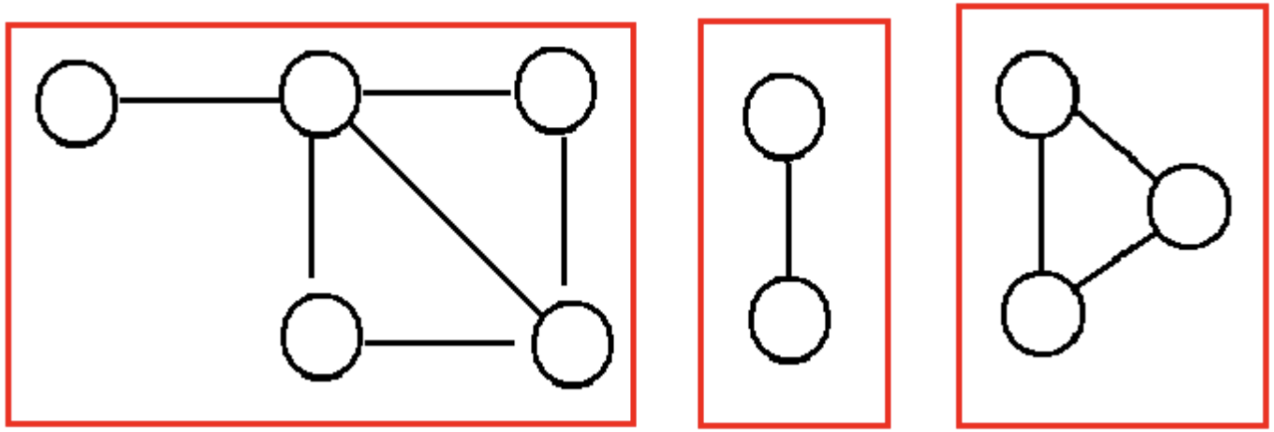
- Graph structures are used to isolate interesting or important sections of a graph
- Structural metrics provide a measurement of a structural property of a graph
  - Global metrics refer to a whole graph
  - Local metrics refer to a single node in a graph
- Identify interesting sections of a graph
  - Interesting because they form a significant domain-specific structure, or because they significantly contribute to graph properties
- A subset of the nodes and edges in a graph that possess certain characteristics, or relate to each other in particular ways



More definitions

Connectivity

- a graph is connected connected if
  - you can get from any node to any other by following a sequence of edges, OR
  - any two nodes are connected by a path
- A directed graph **is strongly connected** if there is a directed path from any node to any other node.
- Every disconnected graph can be split up into a number of connected connected components **components**.



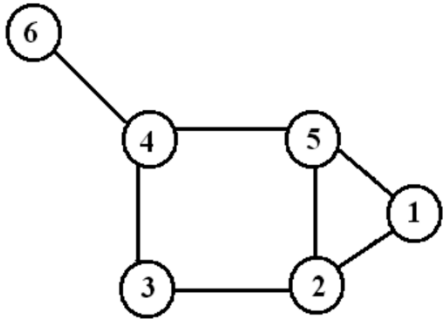
More Definitions

- **Degree**
  - Number of edges incident on a node
- **Degree (directed graphs)**
  - In-degree: Number of edges entering
  - Out-degree: Number of edges leaving
  - Degree = indeg + outdeg
- If G is a graph with with m edges, then

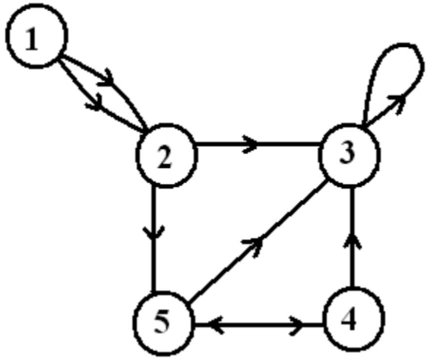
$$\sum \deg(v) = 2m = 2\,|E|$$

- If G is a digraph then

$$\sum \mathrm{indeg}(v) = \sum \mathrm{outdeg}(v) = |E|$$



The degree of 5 is 3



outdeg(1)=2
indeg(1)=0

outdeg(2)=2
indeg(2)=2

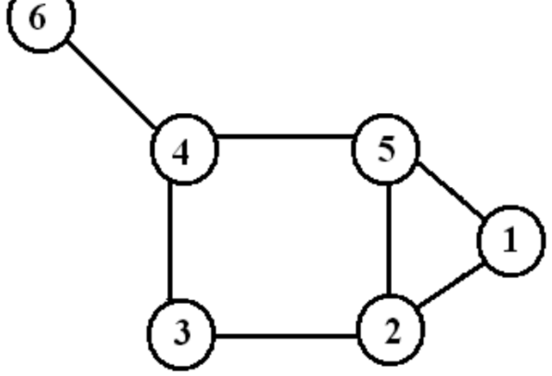
outdeg(3)=1
indeg(3)=4

Graph Traversals - Definitions

- Graph Walks : A walk of length k in a graph is a succession of k edges
- This walk is denoted by u,v,w,x,…,x,z, and is referred to as a walk between u and z
- Closed walk in an undirected graph ends at the starting node
- A walk in a digraph, G, is
  - an alternating sequence of vertices and edges that begins with a vertex, ends with a vertex, and
  - such that for every edge $< u \ \rightarrow v >$ in the walk, vertex u is the element just before the edge, and vertex v is the next element

$$\mathbf{v} ::= v_0 \ \langle v_0 \rightarrow v_1 \rangle \ v_1 \ \langle v_1 \rightarrow v_2 \rangle \ v_2 \ \ldots \ \langle v_{k-1} \rightarrow v_k \rangle \ v_k$$

where $< v_i \rightarrow v_{i+1} > \in E(G), i \in [0..k)$ The walk is said to start at v_o, to end at v_k, and the length, $|v|$, of the walk is defined to be k.

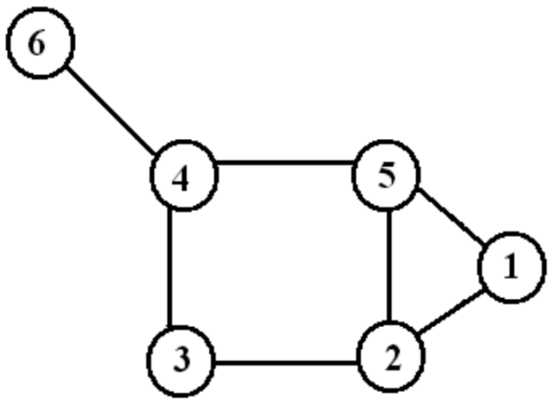


1,2,5,2,3,4
walk of length 5

1,2,5,2,3,2,1
CW of length 6

Graph Traversals - Definitions

- A **path** is a walk in which all the edges and all the nodes are different
  - The shortest walk from one vertex to another is a path
  - The distance*, dist(u,v)*, in a graph from vertex u to vertex v is the length of a shortest path from u to v.



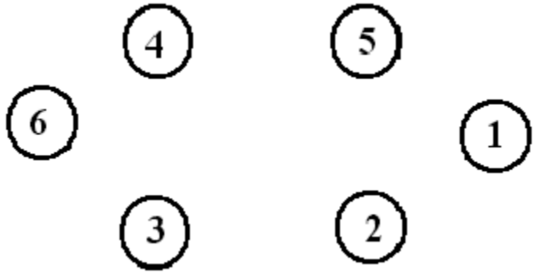
1,2,3,4,6
path of length 4

- A **cycle** is a closed path in which all the edges are different.

1,2,5,1
3-cycle

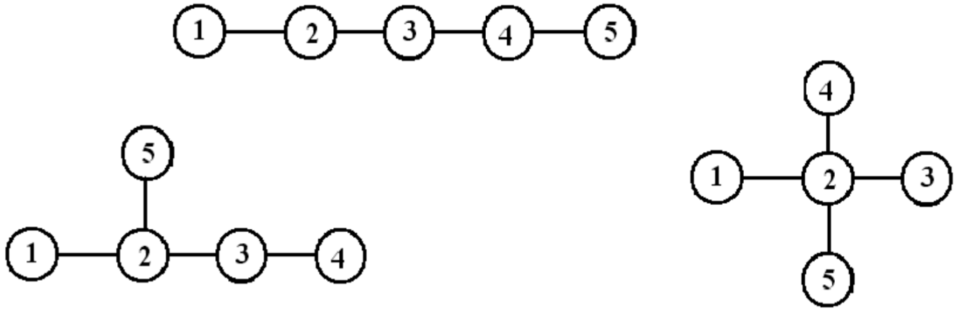
2,3,4,5,2
4-cycle

- Empty Graph / Edgeless graph – No edge

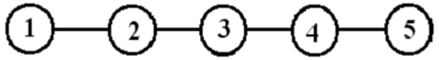


- Null Graph – no nodes, hence no edges

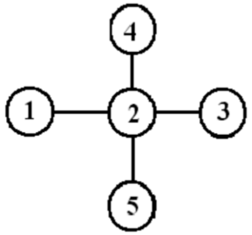
Definitions



- Connected Acyclic Graph
- Two nodes have exactly one path between them

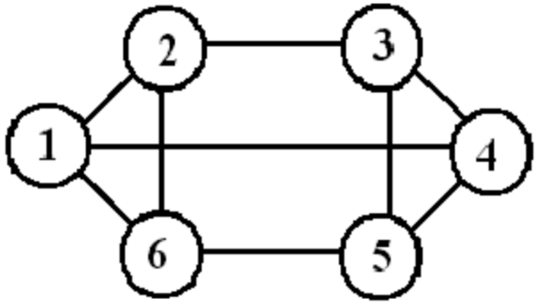


Paths

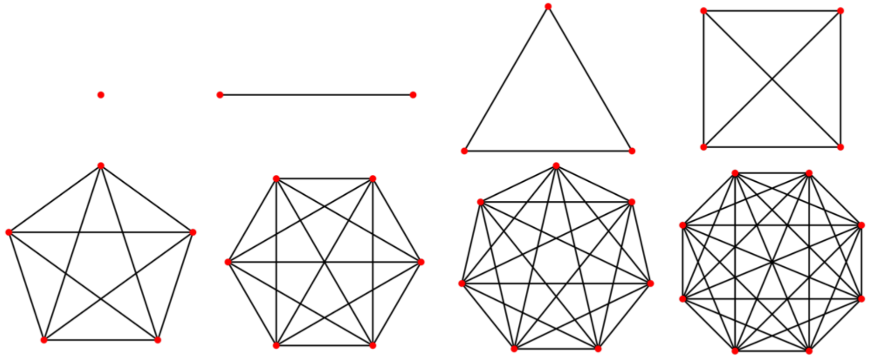


Stars

- Connected Graph: All nodes have the same degree

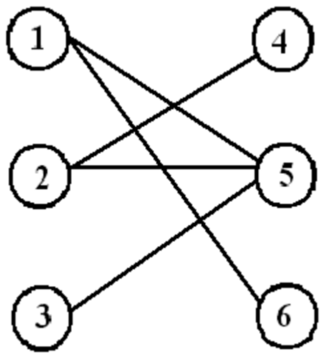


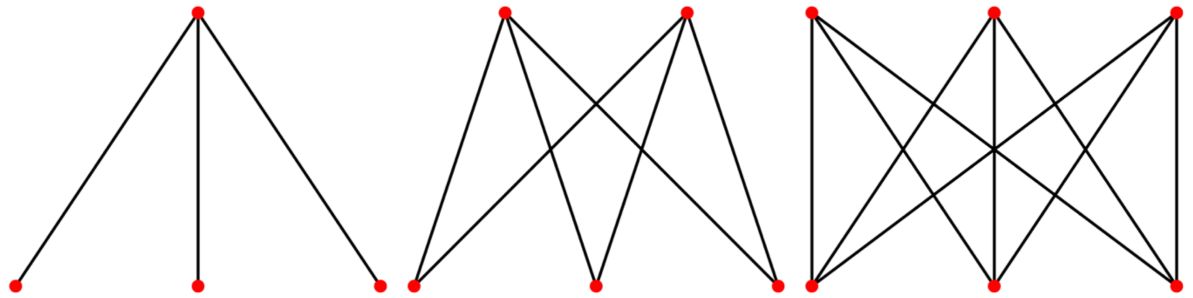
- Complete Graph: Every pair of vertices are adjacent
  - Has $\frac{n(n-1)}{2}$ edges



Bipartite graph

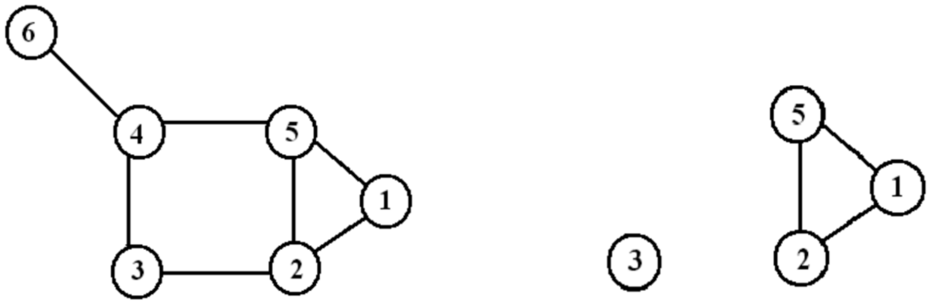
- V can be partitioned into 2 sets V1 and V2 such that $(u, v) \in E$ implies
  - Either $u \in V_1$ and $v \in V_2$
  - Or v $\in V_1$ and $u \in V_2$
- Complete Bipartite graph:
  - Every node of one set is connected to every other node on the other set



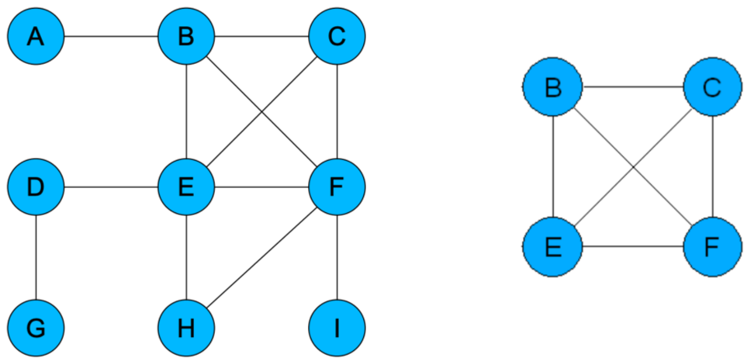


Subgraphs

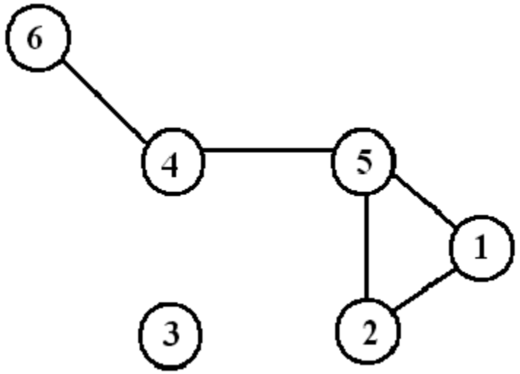
- Vertex and edge sets are subsets of those of G
  - a supergraph of a graph G is a graph that contains G as a subgraph.



- A **clique** is a maximum complete connected subgraph.

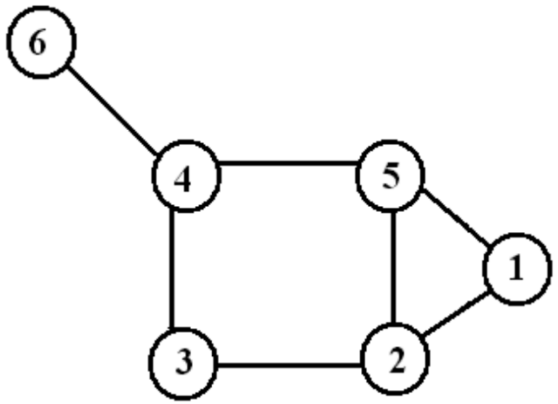


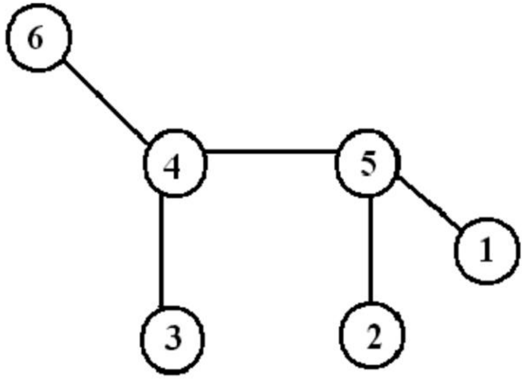
- Spanning Subgraph: H has the same vertex set as G, but possibly not all edges, “H spans G”



Spanning Tree

- Let G be a connected graph.
  - Then a spanning tree spanning tree in G is a subgraph G that includes every node and is also a tree.

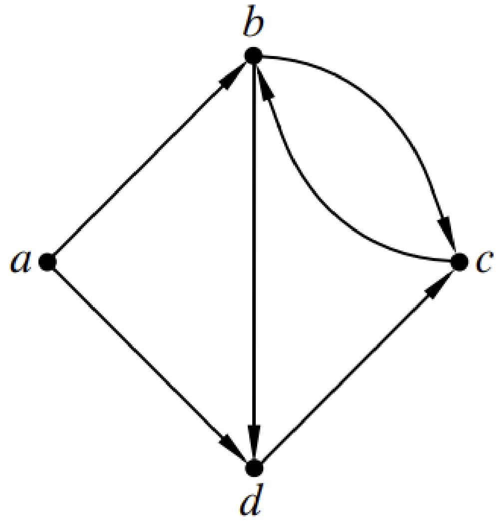




Graph Representation

- Adjacency Matrix
  - V x V
  - Boolean values (adjacent or not)
  - Or Edge Weights

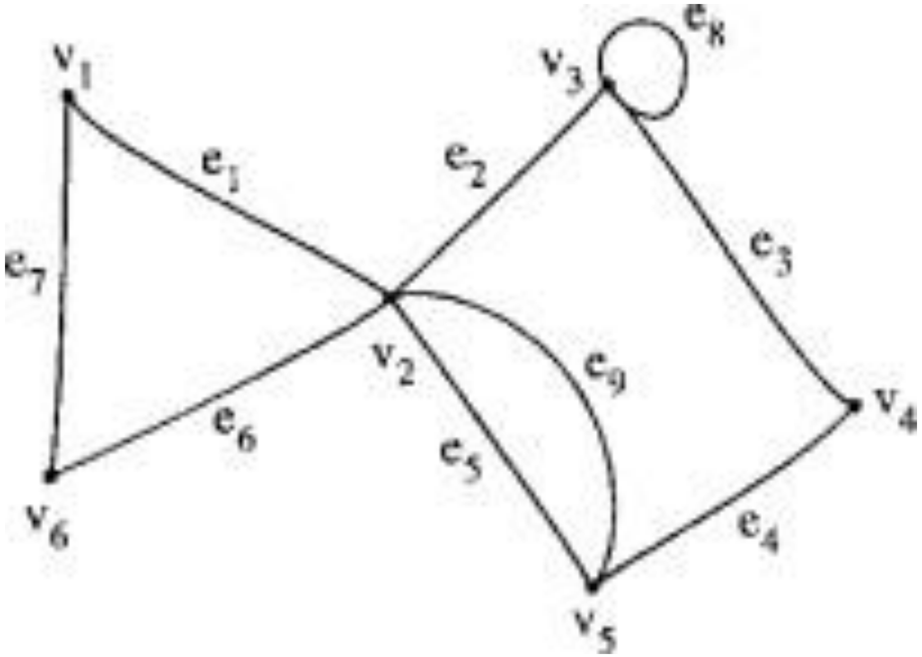
$$(A_G)_{ij} ::= \begin{cases} 1 & \text{if } \langle v_i \to v_j \rangle \in E(G), \\ 0 & \text{otherwise.} \end{cases}$$



$$A_H = \begin{array}{c|cccc|} & a & b & c & d \\ \hline a & 0 & 1 & 0 & 1 \\ b & 0 & 0 & 1 & 1 \\ c & 0 & 1 & 0 & 0 \\ d & 0 & 0 & 1 & 0 \end{array}$$

Graph Representation

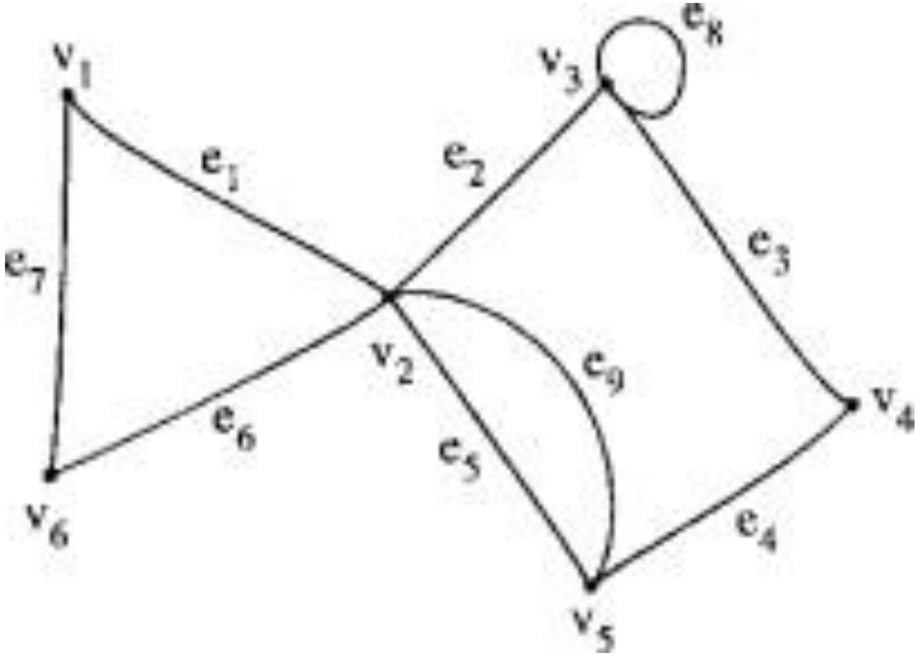
The incidence matrix of a graph G is a $|V| \times |E|$ matrix. The element a_{ij}= the number of times that vertex v_i is incident with the edge e_j.



$$
\begin{array}{c c}
 & \begin{array}{ccccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 & e_9 \end{array} \\
\begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} &
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{pmatrix}
\end{array}
$$

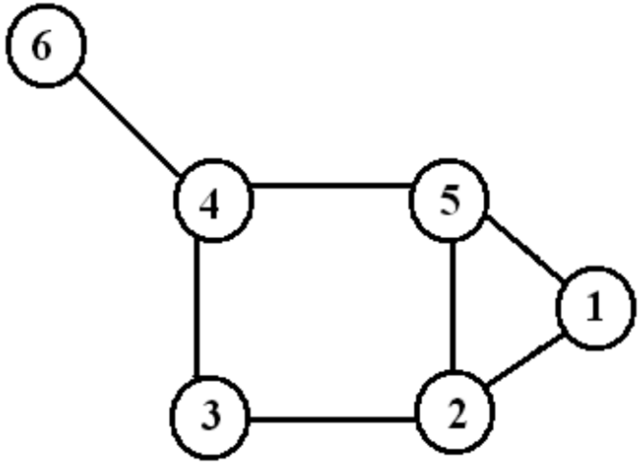
Graph Representation

- Create an adjacency matrix for



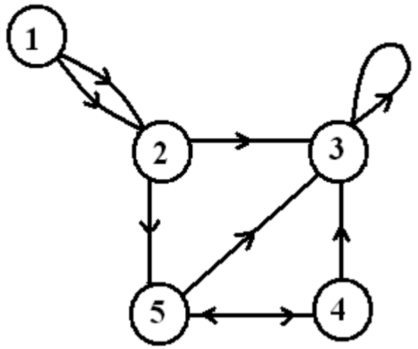
Graph Representation

- Distance Matrix: d_{ij} such that d_{ij} is the topological distance between i and j.



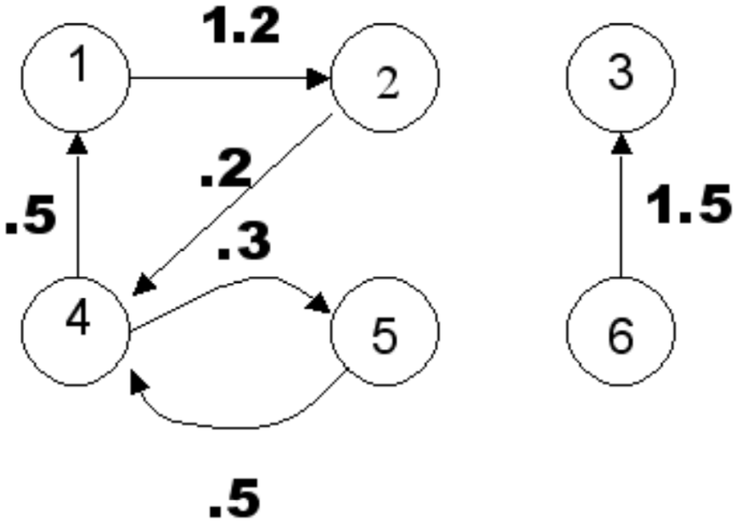
$$\begin{array}{c|cccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
1 & 0 & 1 & 2 & 2 & 1 & 3 \\
2 & 1 & 0 & 1 & 2 & 1 & 3 \\
3 & 2 & 1 & 0 & 1 & 2 & 2 \\
4 & 2 & 2 & 1 & 0 & 1 & 1 \\
5 & 1 & 1 & 2 & 1 & 0 & 2 \\
6 & 3 & 3 & 2 & 1 & 2 & 0
\end{array}$$

Edge List and Node List



Edge List
1 2
1 2
2 3
2 5
3 3
4 3
4 5
5 3
5 4

Node List
1 2 2
2 3 5
3 3
4 3 5
5 3 4

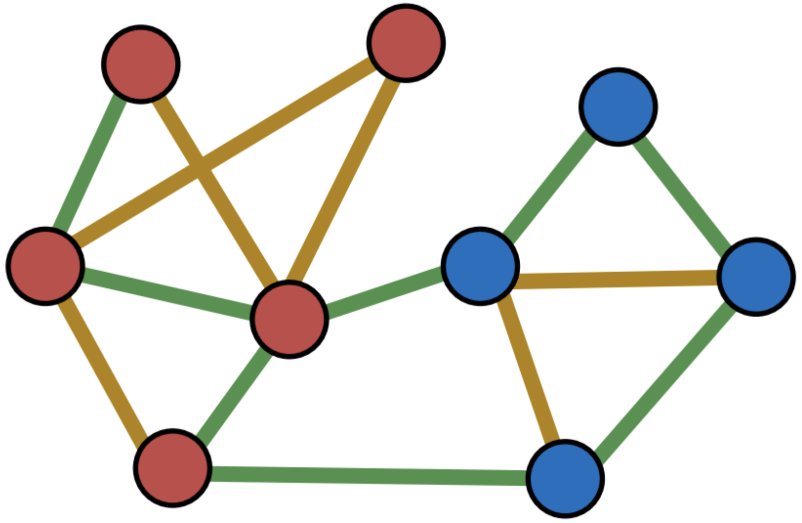


<u>Edge List</u>
1 2 1.2
2 4 0.2
4 5 0.3
4 1 0.5
5 4 0.5
6 3 1.5

Heterogenous Graphs

8 possible relation types!

(Paper, Cite, Paper)
(Paper, Like, Paper)
(Paper, Cite, Author)
(Paper, Like, Author)



(Author, Cite, Author)
(Author, Like, Author)
(Author, Cite, Paper)
(Author, Like, Paper)

Relation types: (node_start, edge, node_end)

- We use **relation type to describe an edge** (as opposed to edge type)
- Relation type better captures the interaction between nodes and edges

Heterogenous Graphs

- **A heterogeneous graph is defined as**

$$G = (V, E, \tau, \phi)$$

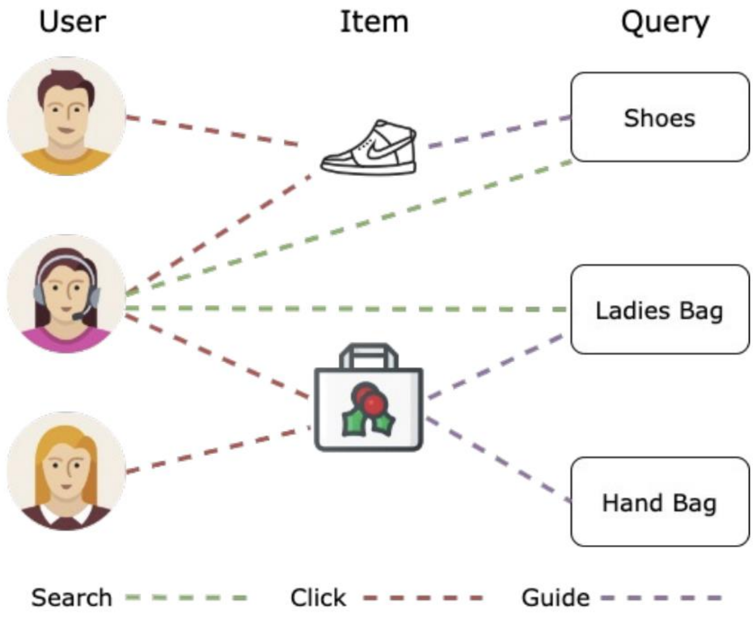
  - Nodes **with node types** $v \in V$
    - **Node type** for node v: $\tau(v)$
  - Edges **with edge types** $(u, v) \in E$
    - **Edge type** for edge (u, v): $\phi(u, v)$
    - **Relation type** for edge e is a tuple: $r(u, v) = (\tau(u), \phi(u, v), \tau(v))$

An edge can be described as a pair of nodes

- There are other definitions for heterogeneous graphs as well – describe **graphs with node & edge types**

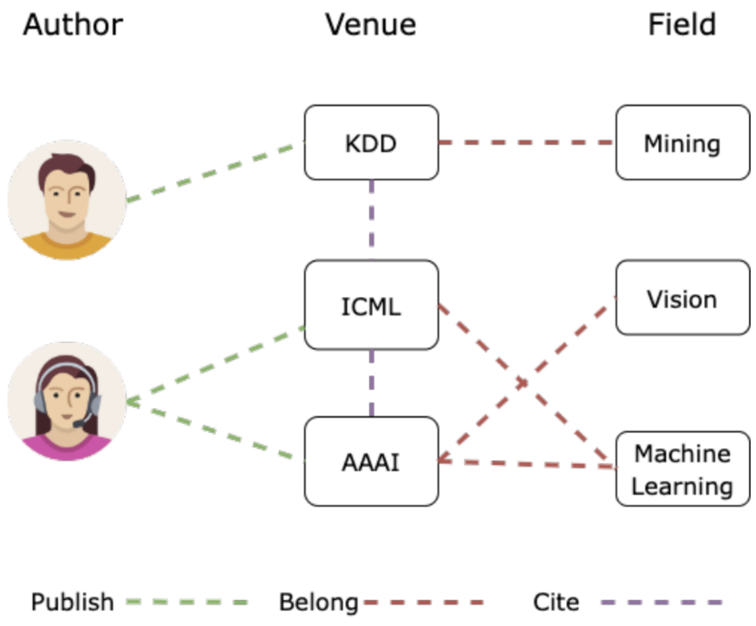
Examples

- Example: E-Commerce Graph
  - **Node types:** User, Item, Query, Location, ...
  - **Edge types:** Purchase, Visit, Guide, Search, ...
  - Different node type's features spaces can be different!



Examples

- Example: Academic Graph
  - **Node types:** Author, Paper, Venue, Field, ...
  - **Edge types:** Publish, Cite, ...
  - Benchmark dataset: **Microsoft Academic Graph**

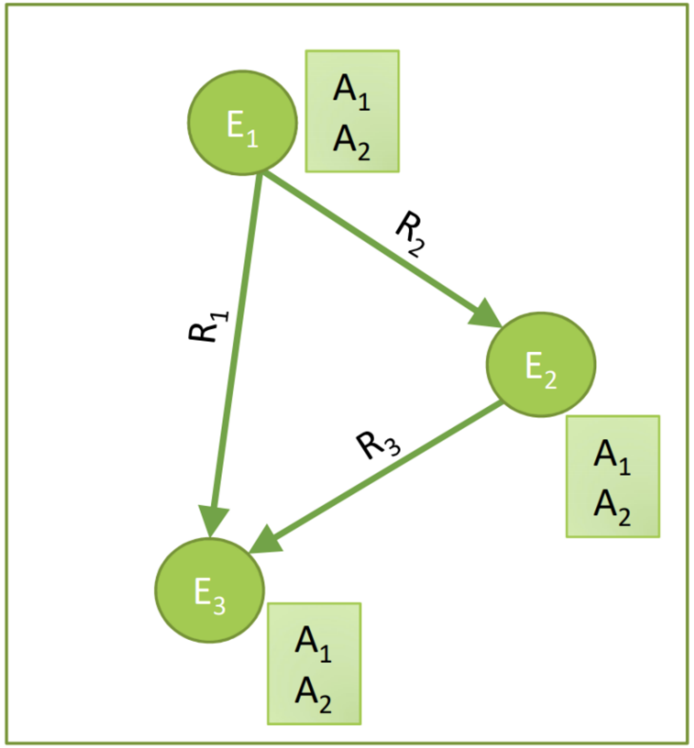


Knowledge Graphs

Knowledge in graph form:

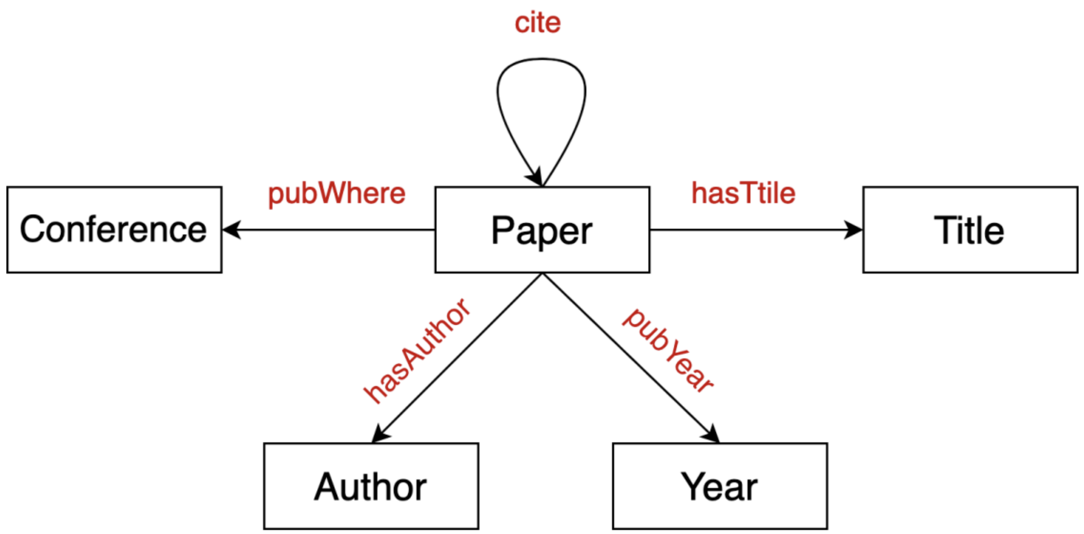
- Capture entities, types, and relationships

- Nodes are **entities**
- Nodes are labeled with their **types**
- Edges between two nodes capture **relationships** between entities
- **KG is an example of a heterogeneous graph**



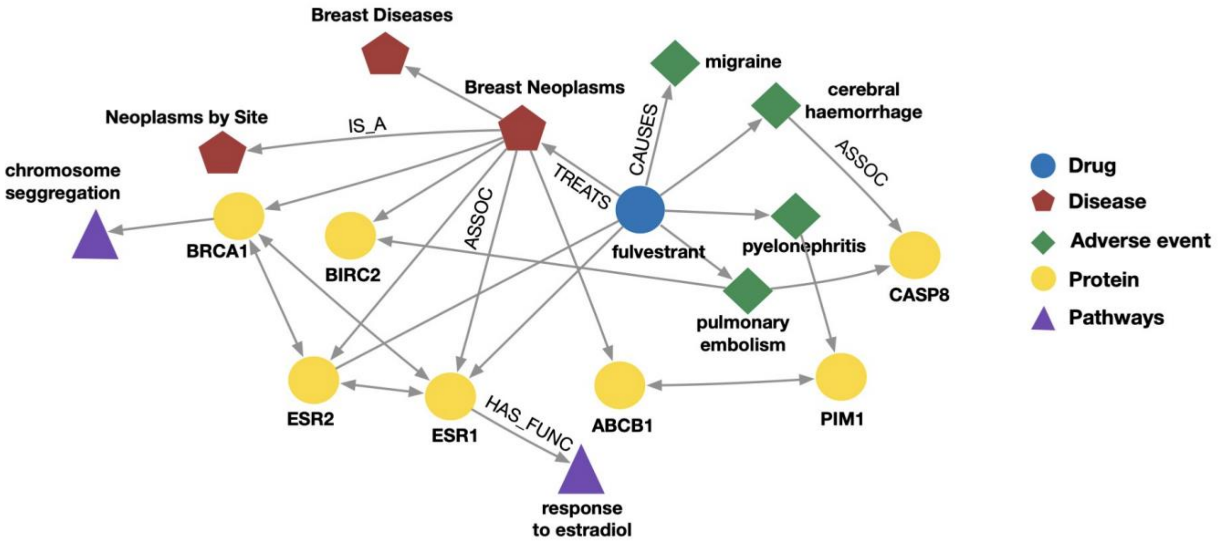
Scholarly Data

- **Node types**: paper, title, author, conference, year
- **Relation types**: pubWhere, pubYear, hasTitle, hasAuthor, cite



Bio- Knowledge Graphs

- **Node types**: drug, disease, adverse event, protein, pathways
- **Relation types**: has_func, causes, assoc, treats, is_a

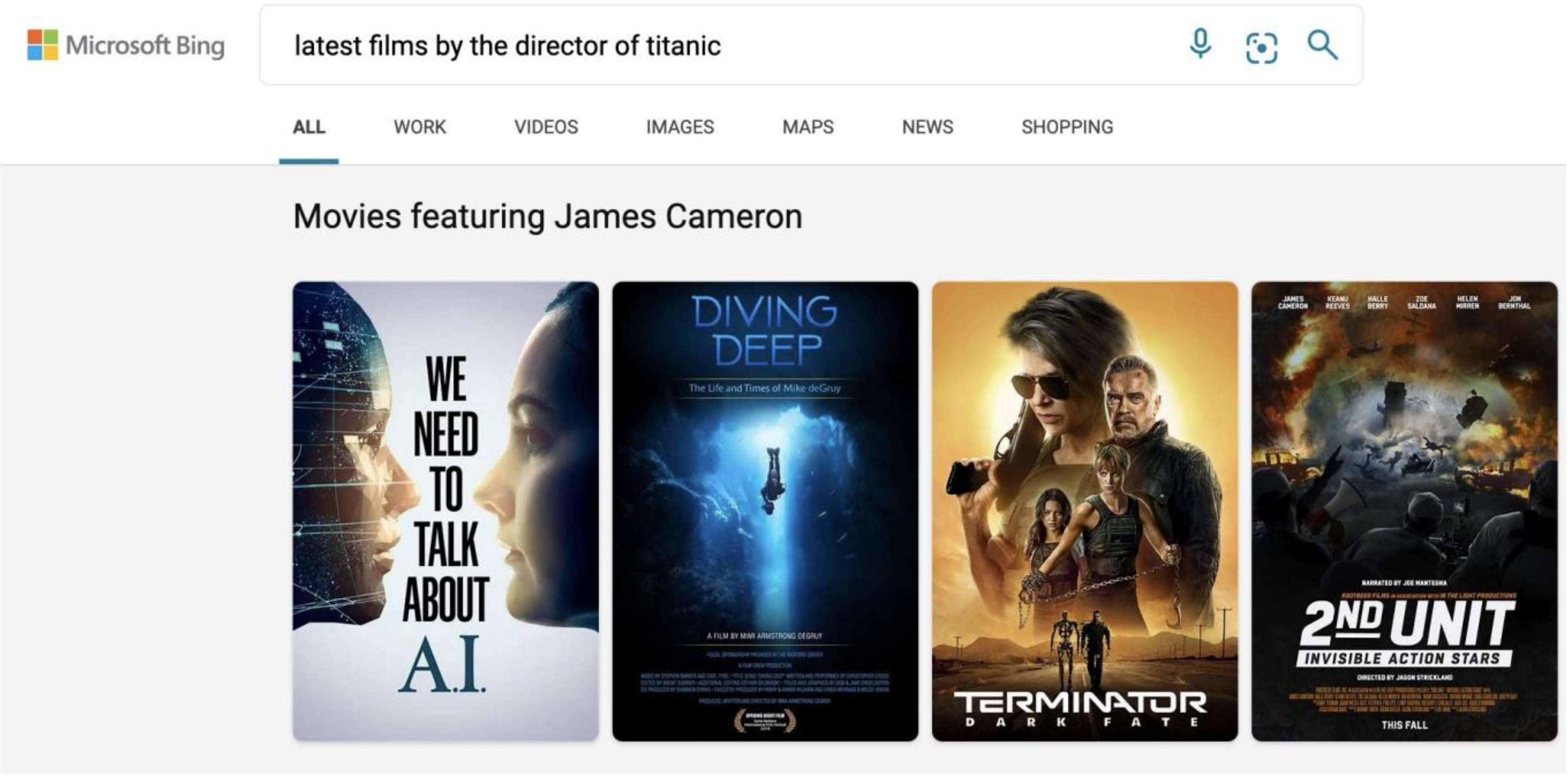


Knowledge Graphs

Examples of knowledge graphs

- Google Knowledge Graph
- Amazon Product Graph
- Facebook Graph API
- IBM Watson
- Microsoft Satori
- Project Hanover/Literome
- LinkedIn Knowledge Graph
- Yandex Object Answer

Applications



Literature

- https://www.cs.mcgill.ca/~wlh/grl_book/
- https://ocw.mit.edu/courses/6-042j-mathematics-for-computer-science-spring-2015/mit6_042js15_session16.pdf