

# Machine Learning for Graphs and Sequential Data

## *Graphs – Graphs & Networks*

Lecturer: Prof. Dr. Stephan Günnemann

[cs.cit.tum.de/daml](https://cs.cit.tum.de/daml)

---

Summer Term 2024

Data Analytics and  
Machine Learning



# Roadmap

---

- **Chapter: Graphs**

- 1. Graphs & Networks**

- **Motivation & Definitions**
    - Properties of Real Networks

- 2. Generative Models

- 3. Ranking

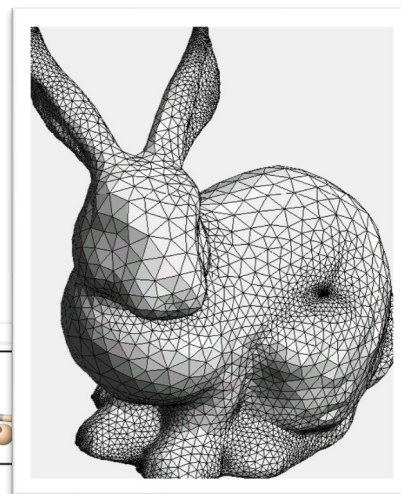
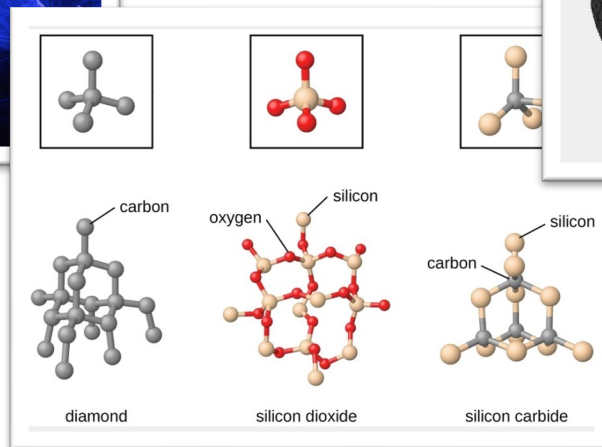
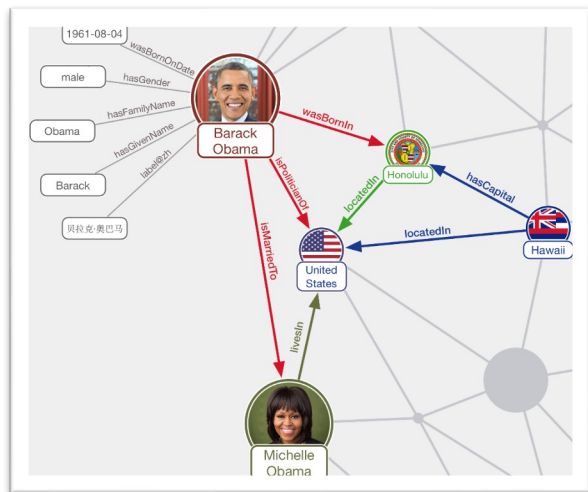
- 4. Clustering

- 5. Classification (Semi-Supervised Learning)

- 6. Node/Graph Embeddings

- 7. Graph Neural Networks (GNNs)

# Why Should We Care?



- Web: hyper-text graph
- Information retrieval: bi-partite graphs (documents-terms)
- E-commerce: User-Product graphs

# Why Should We Care?

---

- ‘Viral’ marketing, News propagation
- Computer network security: email/IP traffic and anomaly detection
- Ranking of search results
- Fraud detection in e-commerce systems
- Drug discovery, molecule property prediction
- Scene graph analysis
- ...

# Basic Definition

---

- Plain/simple graph  $G = (V, E)$ 
  - Set of nodes  $V$
  - Set of edges  $E \subseteq V \times V$       // for undirected graphs:  $(i, j) \in E \Leftrightarrow (j, i) \in E$
- Equivalent representation via (binary) adjacency matrix  $A \in \{0,1\}^{|V| \times |V|}$
- Multiple extensions possible
  - weighted graphs (node weights, edge weights)
  - attributed graphs (multi-dimensional vectors assigned to nodes/edges)
  - temporal graphs (timestamp associated with node/edge)

# Roadmap

---

- **Chapter: Graphs**

- 1. Graphs & Networks**

- Motivation & Definitions
    - **Properties of Real Networks**

- 2. Generative Models

- 3. Ranking

- 4. Clustering

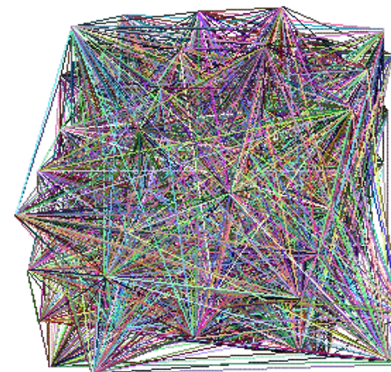
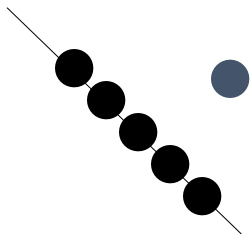
- 5. Classification (Semi-Supervised Learning)

- 6. Node/Graph Embeddings

- 7. Graph Neural Networks (GNNs)

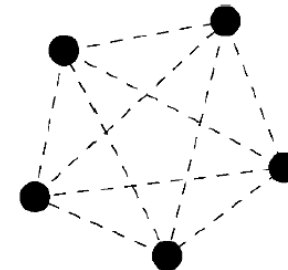
# How are real networks structured?

- What does the Internet look like?
- What does Facebook look like?
- What is 'normal'/'abnormal'?
- Which patterns/laws hold?
  - To spot anomalies (rarities), we have to discover patterns
  - **Large datasets** reveal patterns/anomalies that may be invisible otherwise...

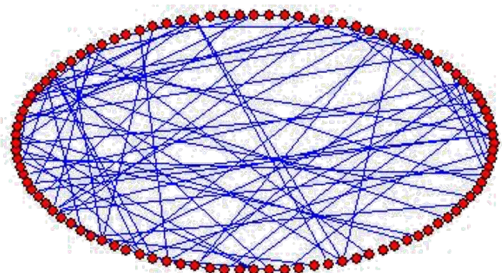


# Are real graphs random?

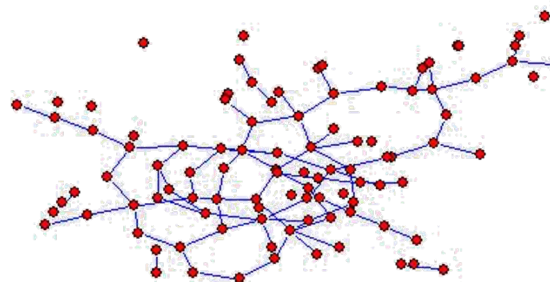
- Erdős-Renyi Random Graph Model
  - Start with  $N$  (isolated) nodes
  - For every pair  $v_1, v_2 \in V$  add an edge with probability  $p$
  - Every edge occurs with equal probability
- Example: 100 nodes, avg degree = 2



*before layout*



*after layout*



- No obvious patterns



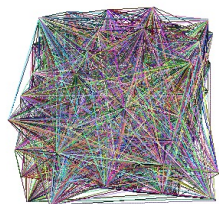
# Laws and Patterns

---

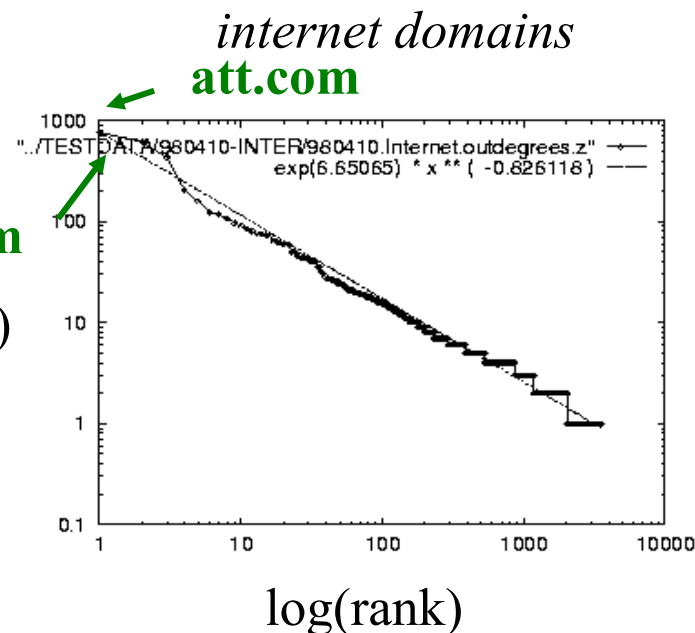
- Q: Are real graphs random?
- A: NO!
  - Diameter
  - in- and out- degree distributions
  - other (surprising) patterns
  
- So, let's look at the data

# Power Law Distributions

- Gaussian distributions are common in nature
- In networks, however, a **power law distribution** often explains the data better
- Example: Power law in the degree distribution



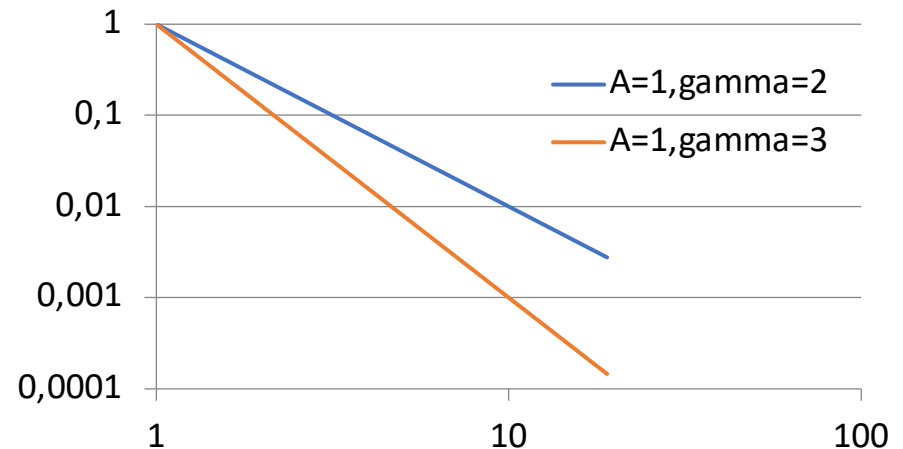
**ibm.com**  
log(degree)



# Power Law Distributions

- Definition Power Law:
  - Two variables  $x$  and  $y$  are related by a power law when  $y(x) = Ax^{-\gamma}$  where  $A$  and  $\gamma$  (power law exponent) are positive constants
  - A random variable is distributed according to a power law when the probability density function (pdf) is given by  $p(x) = Ax^{-\gamma}$  with  $\gamma > 1$

- Note: Power law distribution looks like a line on a log-log scale
- **Characteristic: Decay of pdf is only polynomial (Gaussian: exponential)**
- → More likely to observe values far to the right of the mean

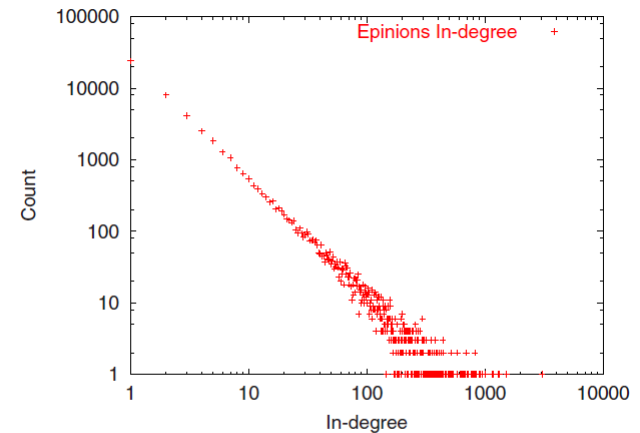


*a.k.a. heavy-tailed distributions*

- E.g. more likely to have nodes with a very high degree

# Examples: Power Law Distributions

- "Internet AS" graph with exponent 2.1 – 2.2
- "Internet router" graph with exponent 2.48
- Citation graphs with exponent 3
- Epinions (who-trust-whom)
- ...

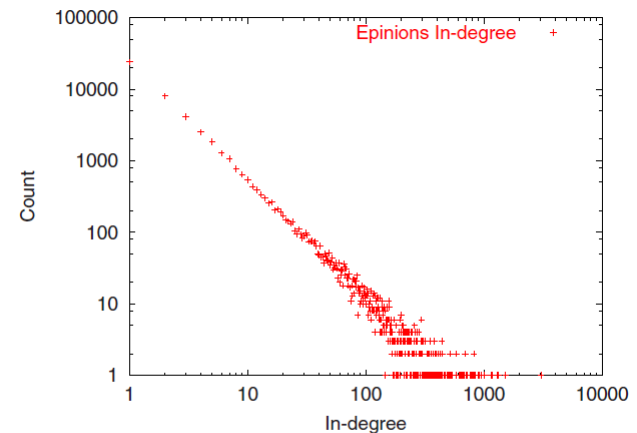


(a) Epinions In-degree

- Note: Graphs with degree distributions following a power law are called **scale-free**
  - $y(a \cdot x) = b \cdot y(x)$  //  $y(x)$ =number of nodes with degree  $x$

## Important Remark

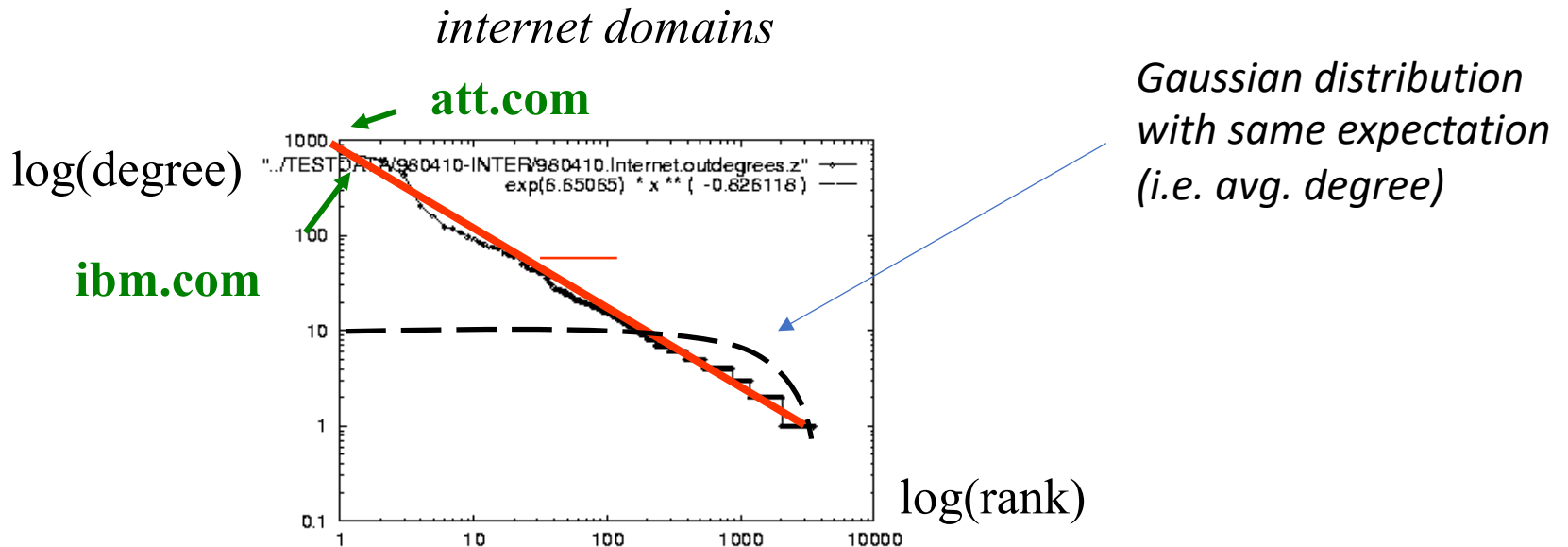
- Important: We do not claim that the data's underlying distribution is a power law distribution
- Instead: In many cases a power law distribution is a **good description/fit** (or approximation) of what we observe
  - usually deviations to power law distribution are observed (to different degrees)
  - other models: exponential cutoff, lognormal distributions



(a) Epinions In-degree

# "Gaussian Trap"

- Q: So what?
- A1: Be careful when writing algorithms!
  - Example: # of two-step-away pairs (= friends of friends)
    - $O(d_{\max}^2) \sim 10M^2$   
for storage:  $\sim 0.8\text{PB} \rightarrow$  a data center(!)

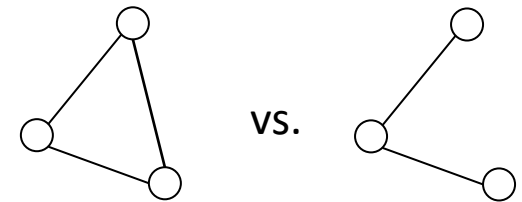


# Patterns and Algorithms

- Q: So what?
- A2: Patterns allow to design new algorithms!

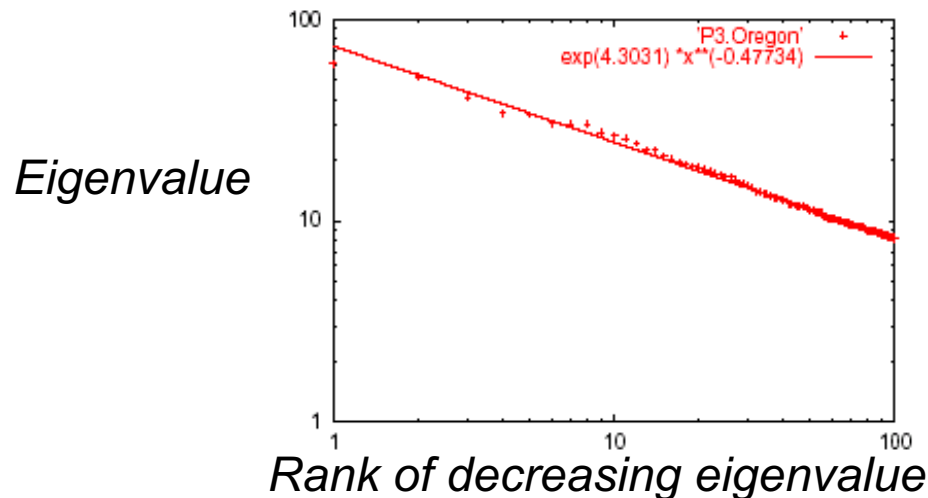
## Example: Clustering Coefficient

- Real social networks have a lot of triangles
  - Friends of friends are friends
- Clustering Coefficient
  - $C = \frac{3 \cdot \text{number of triangles}}{\text{number of connected triplets}}$
  - Real-world networks often show large clustering coefficients  
→ Strong community structure
- Computation of clustering coefficient requires computation of triangles
  - But: **triangles are expensive to compute** (3-way join)



# Efficient Triangle Counting

- How to efficiently estimate the number of triangles?
  - Let's look at some patterns...
- Recap: Adjacency matrix  $\mathbf{A} \in \{0, 1\}^{N \times N}$ 
  - $a_{ij} = 1$  if there is an edge between  $i$  and  $j$ , 0 otherwise
- Recap: Eigenvalue decomposition  $\mathbf{A} \mathbf{x} = \lambda \mathbf{x}$
- Observer: Power law in the eigenvalues of the adjacency matrix



Exponent = slope

$$E = -0.48$$



# Efficient Triangle Counting

- How does this help?
- 1. Some nice fact (easy to show):
  - number of triangles  $= \frac{1}{6} \text{trace}(A^3) = \frac{1}{6} \sum_i \lambda_i^3$
  - $\lambda_i$  = eigenvalues of adjacency matrix A
- 2. Eigenvalues follow power law (highly skewed)
  - we only need the top few (largest) eigenvalues!
  - how can we compute them efficiently?

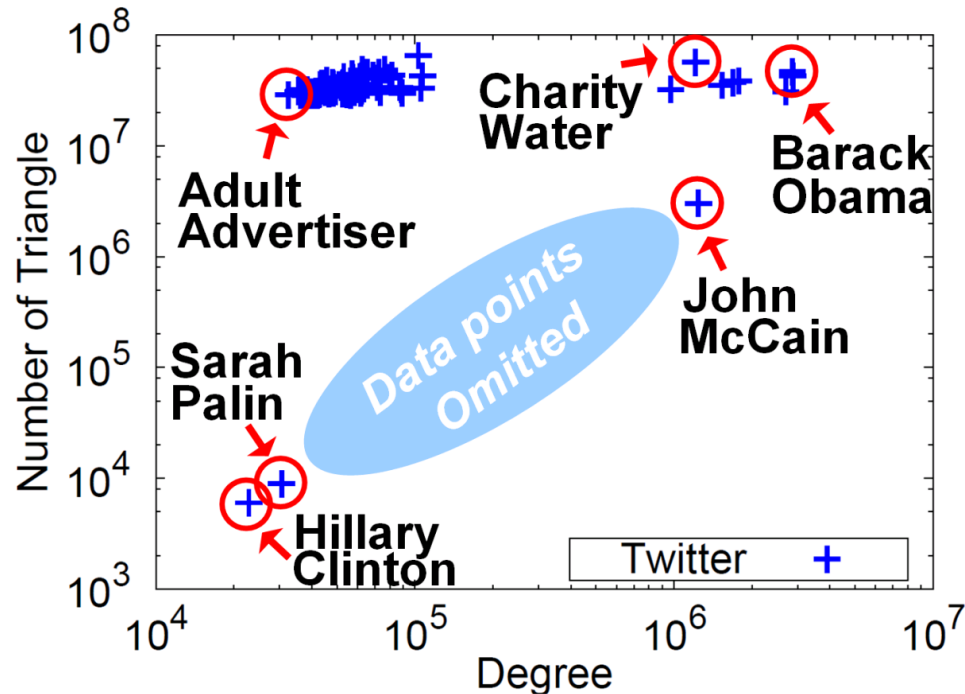
# Recap: Power Iteration

- Eigenvalues are important for many ML/data mining tasks
  - PCA, Ranking of Websites, Community Detection, ...
  - How to compute them efficiently?
- Power iteration (a.k.a. Von Mises iteration)
  - Iterative approach to compute **a single** eigenvector
- Let  $X$  be a matrix and  $v$  be an arbitrary (normalized) vector
  - Iteratively compute  $v \leftarrow \frac{X \cdot v}{\|X \cdot v\|}$  until convergence
    - in each step,  $v$  is simply multiplied with  $X$  and normalized
  - **$v$  converges to the eigenvector of  $X$  with greatest absolute value**
  - Highly efficient for sparse data

# Recap: Power Iteration

- Convergence:
  - Linear convergence with rate  $|\lambda_2/\lambda_1|$
  - Fast convergence if first and second eigenvalue are dissimilar
  
- How to find **multiple (the k largest) eigenvectors**?
  - Let us focus on symmetric matrices  $X$
  - Eigenvalue decomposition leads to:  $X = \Gamma \cdot \Lambda \cdot \Gamma^T = \sum_{i=1}^d \lambda_i \cdot \gamma_i \cdot \gamma_i^T$
  - Define deflated matrix:  $\hat{X} = X - \lambda_1 \cdot \gamma_1 \cdot \gamma_1^T$ 
    - $\hat{X}$  has the same eigenvectors as  $X$  except the first one has become zero
  - Apply power iteration on  $\hat{X}$  to find the second largest eigenvector of  $X$

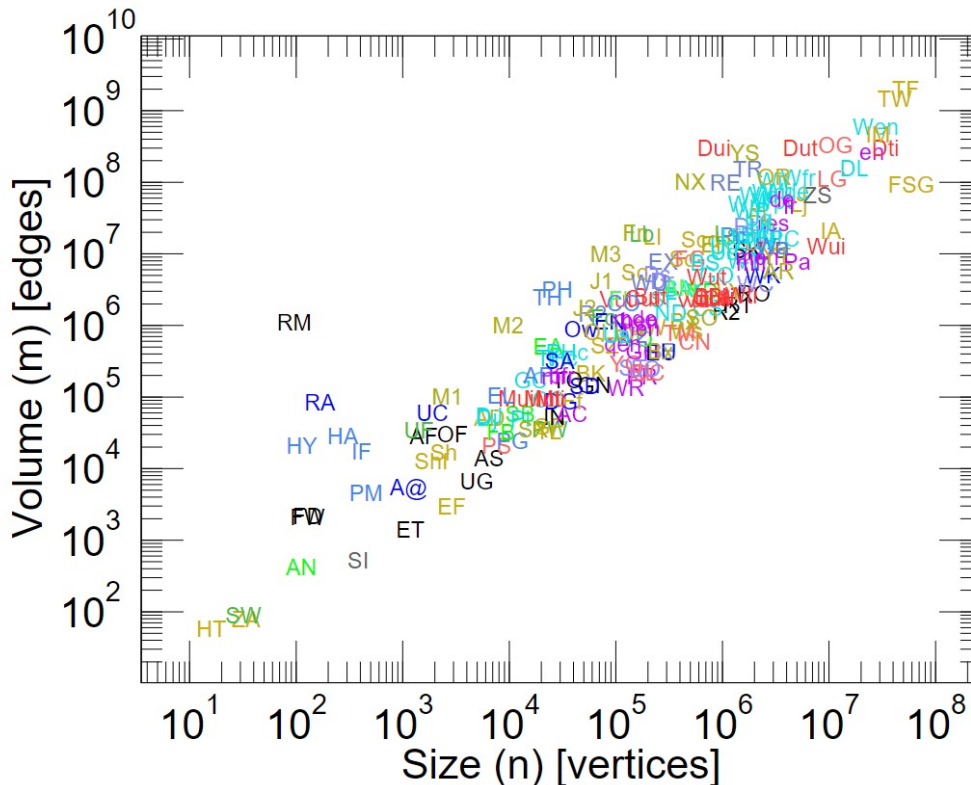
## Example: Analysis of Twitter



- Anomalous nodes in Twitter (~ 3 billion edges)
- [U Kang et. al., PAKDD'11]

# Real Graphs are Sparse

- $N^2$  possible edges for a graph with  $N$  nodes
- However, real-world graphs are very sparse  $E \ll N^2$
- Instead of  $E = O(N^2)$ , we see  $E = O(N^\alpha)$  with  $\alpha$  significantly less than 2



Every “XX” is a real world network

Note the log-log scale

$$\alpha \approx 1.4$$

# Small World Phenomenon

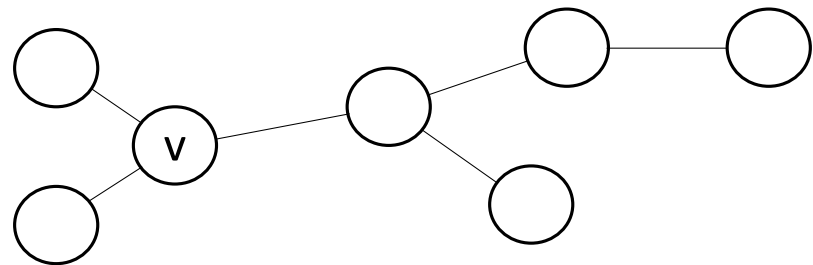
- Famous experiment by Travers and Milgram [TM1969]
  - Setup: Try to reach a random person by sending a chain letter
  - Result: The average length of chains that reach the person was six
  - → Length very small compared to number of participants
  - → „Small world phenomenon“; „six degrees of separation“
- Ways to measure this phenomenon
  - Characteristic path length
  - Average diameter
  - Effective diameter/Eccentricity



# Characteristic Path Length & Average Diameter

## ■ Characteristic path length

- For each starting node  $v \in V$  consider the shortest path to every other node
- Take the average length of all these paths
- Consider average path length for all starting nodes and take the median
- $\text{median}_{v \in V} \left\{ \frac{1}{|V|} \sum_{v_j \in V} \text{len}(p_{\min}(v, v_j)) \right\}$

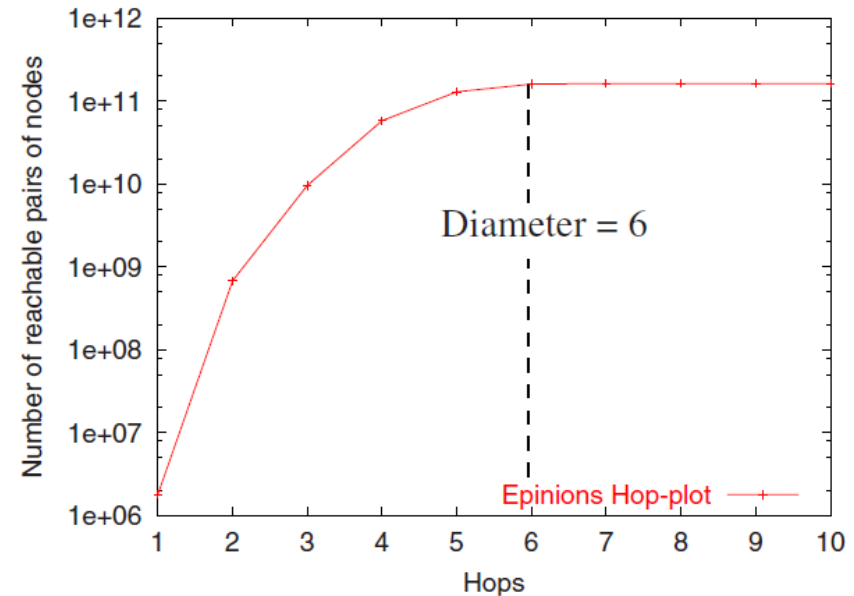


## ■ Average diameter

- Similar to above but use (in last step) the mean instead of the median
- $\frac{1}{|V|} \sum_{v \in V} \frac{1}{|V|} \sum_{v_j \in V} \text{len}(p_{\min}(v, v_j))$

# Effective Diameter / Eccentricity & Hop-plot

- Let  $N_h(u)$  be the number of nodes reachable from  $u$  via  $h$  hops
  - $N_h(u) = \{v \in V \mid \text{len}(p_{\min}(u, v)) \leq h\}$
- The total neighborhood size  $N_h$  is the sum over all starting nodes
  - $N_h = \sum_{u \in V} |N_h(u)|$
- Hop-plot: Plot of  $N_h$  versus  $h$
- Effective diameter (or Eccentricity)
  - Minimum number of hops in which some fraction (e.g. 90%) of all connected pairs of nodes can reach each other
  - $\min\{k \in \mathbb{N} \mid N_k \geq 0.9 \cdot |V|^2\}$
  - **Advantage: Also works for disconnected graphs**





# Importance of „Network Laws“

---

- Laws describing „normal“ networks are important for:
- Design of algorithms
- Detection of abnormal/interesting patterns
  - Abnormalities deviate from the „normal“ patterns
  - Prerequisite: specify what is normal
- Development of graph generators
  - Often: real world data not public available; or just small excerpts
  - Use synthetic data to test algorithms
  - Requirement: generate synthetic but realistic graphs
- Simulation studies
  - E.g. test next-generation internet protocol on graph „similar“ to what Internet will look like a few years into the future

# Questions

---

- How much memory do you need to store the edges of a graph with 1000 nodes and 10,000 edges in a dense adjacency matrix? How much for a sparse matrix?
- What is the average degree in an Erdős-Renyi graph with edge probability  $p$ ? And in a real world sparse graph with  $O(E) = O(N^\alpha)$ ?

# Reading Material

---

- "Graph Mining: Laws, Tools, and Case Studies" by Deepayan Chakrabarti, Christos Faloutsos

# Machine Learning for Graphs and Sequential Data

## *Graphs – Graphs & Networks*

Lecturer: Prof. Dr. Stephan Günnemann

[cs.cit.tum.de/daml](https://cs.cit.tum.de/daml)

---

Summer Term 2024

Data Analytics and  
Machine Learning

