Network Analysis

Marta Arias, UPC

Grau en Enginyeria Informàtica, UPC

November 21, 2025

Characterizing real networks

What is a Network?

- A network[1] is a collection of **nodes** (or vertices) and **edges** (or links) that connect them.
- This simple structure can represent incredibly complex systems.
- We are drowning in relational data, and we need a “network lens” to understand its structure.

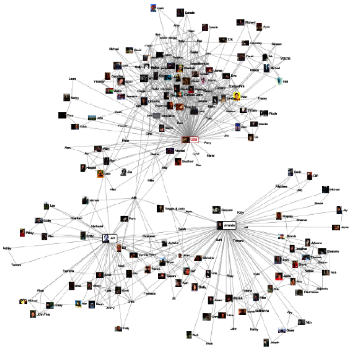
[1] Network = *graph*

Basic terminology

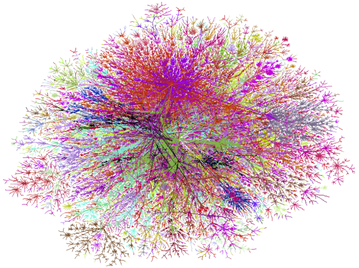
- graph $G = (V, E)$
- set of *nodes* V, set of *edges* $E \subseteq VxV$
- number of nodes $n = |V|$, number of edges $m = |E|$
- can be *weighted/unweighted*
- can be *directed/undirected*
- connections represented by *edge lists* and/or *adjacency matrices*

Examples of Complex Networks

Social Networks: Nodes are people, edges are friendships, follows, or collaborations.

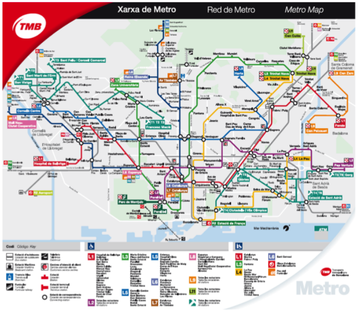


Information Networks: Nodes store information, links associate it.
Examples: *The World Wide Web, citation networks, P2P networks.*

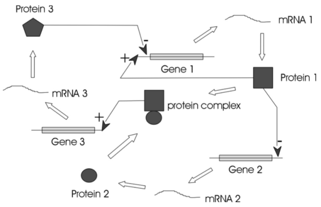


More examples of Complex Networks

Technological Networks:
Designed for transport or communication.
Examples: *The Internet (routers/links), power grids, airline routes.*



Biological Networks:
Representing the building blocks of life.
Examples: *Protein-protein interaction networks, gene regulatory networks, neural networks.*



From Simple to Complex

- **Simple Networks** (like a grid) are regular and predictable.
- **Random Networks** have no “structure” at all.
- **Complex Networks** (like the Web, or a social network) are neither regular nor random. They have non-trivial, emergent properties.

We will focus on two of the biggest “defining qualities” for real-world complex networks.

“Small-World” Networks

- **Observation:** Most real-world networks are “small.”
- **Short Average Path Length:** Any two nodes are connected by a surprisingly short path.
- **High Clustering Coefficient:** Your friends are likely to be friends with each other.
    - This is the “cliquey-ness” of a network.

Measuring connectedness

The Shortest Path (Geodesic)

The path between two nodes, u and v, with the minimum number of edges (in an unweighted graph) or the minimum sum of edge weights (in a weighted graph).

- **Notation:** $d(u, v)$

Eccentricity

The eccentricity $e(v)$ of a node v is its maximum shortest-path distance to any other node u in the graph. It answers: "How far is this node from the node farthest from it?"

- **Formula:** $e(v) = \max_{u \in V} d(v, u)$

Network metrics for connectedness

Network Diameter (D)

The maximum eccentricity of any node in the graph. It represents the “longest shortest path” in the entire network and provides a measure of the network’s overall size.

- **Formula:** $D(G) = \max_{v \in V} e(v)$

Network Radius (R)

The minimum eccentricity of any node in the graph. The node(s) with this eccentricity are the most central.

- **Formula:** $R(G) = \min_{v \in V} e(v)$

Network metrics for connectedness, cont.

Average Shortest Path Length (ASPL)

The average of the shortest-path distances over all unique pairs of nodes in the graph. It provides a global measure of network efficiency and compactness.

- **Formula:** $L(G) = \frac{2}{n(n-1)} \sum_{u<v} d(u, v)$
- Above formula problematic if G not connected
    - use *harmonic mean* instead, or
    - restrict metric to connected components

Graph Center

The set of all nodes whose eccentricity is equal to the radius $(e(v) = R(G))$. These are the most central nodes.

Graph Periphery

The set of all nodes whose eccentricity is equal to the diameter $(e(v) = D(G))$. These are the nodes at the "edge" of the network.

Are real networks well connected?

Despite often having millions or billions of nodes, the average distance between any two nodes (the average path length) is **surprisingly short**.

- Milgram's "Six Degrees" (Milgram 1967)

  The foundational experiment. Letters sent across the US reached their target in a median of just 6 steps, coining the phrase "six degrees of separation.

- MSN Messenger (Leskovec and Horvitz 2008)

  A study by Leskovec & Horvitz of 240 million users showed a global average path length of 6.6. This was the first large-scale digital confirmation of Milgram's theory.

- Facebook "Four Degrees" (Backstrom et al. 2012)

  A study of 721 million users found the average path length had shrunk to 4.74. This showed "four degrees of separation" on a global scale, a number that continues to decrease.

- Biological & Tech Networks

  The C. elegans worm's neural map ($L=2.65$) (Watts and Strogatz 1998) and the Internet's router-level graph ($L \approx 4$) (Faloutsos, Faloutsos, and Faloutsos 1999) are also extremely small, efficient networks.

Defining Clustering Coefficient

The **Clustering Coefficient** (C) is a fundamental measure of network structure.

- It quantifies the “transitivity” of a network.
- It measures the degree to which nodes in a graph tend to cluster together.
- The guiding question: “What fraction of my neighbors are also neighbors with each other?”

We measure this in two primary ways:

1. **Local Clustering Coefficient:** For a single node.
2. **(Global) Clustering Coefficient:** For the entire network.

Local Clustering Coefficient (C_i)

The local clustering coefficient of a *single node* `i` is the fraction of its neighbors that are also connected to each other.

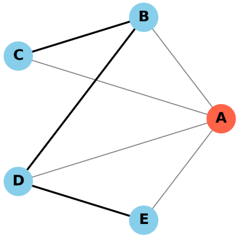
Formula:

$$C_i = \frac{\text{Number of connections between } i\text{'s neighbors}}{\text{Total possible connections between } i\text{'s neighbors}}$$

$$C_i = \frac{2E_i}{k_i(k_i - 1)}$$

- k_i is the degree of node `i` (its number of neighbors).
- E_i is the number of edges that *actually exist* between the neighbors of `i`.

Local Clustering: Example



For Node A:

- Neighbors: degree $k_A = 4$
- Possible connections between neighbors: $\frac{k_A(k_A-1)}{2} = \frac{4(3)}{2} = 6$
- Actual connections between neighbors: 3 (B-C, B-D, D-E)
- **Local Clustering:** $C_A = \frac{3}{6} = 0.5$

(Global) Clustering Coefficient / Transitivity

This measures the clustering of the *entire network*. There are two common ways to define it:

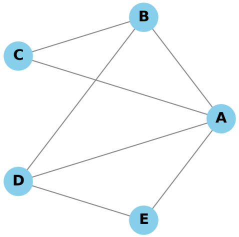
A. Average Local Clustering Coefficient ($\bar{C}$)

- The simplest way: just calculate the average of all the individual, local clustering coefficients.
- **Formula:** $\bar{C} = \frac{1}{n} \sum_{i=1}^{n} C_i$

B. Transitivity (C_T)

- A more technically precise definition (Watts & Strogatz).
- It's the ratio of "triangles" to "triplets" in the *entire* network.
  - **Triangle:** Three nodes, all connected (A-B, B-C, C-A).
  - **Triplet:** Three nodes with at least two edges (e.g., A-B, B-C).
- **Formula:** $C_T = \frac{3 \times (\text{number of triangles})}{(\text{number of connected triplets})}$

Example of Transitivity

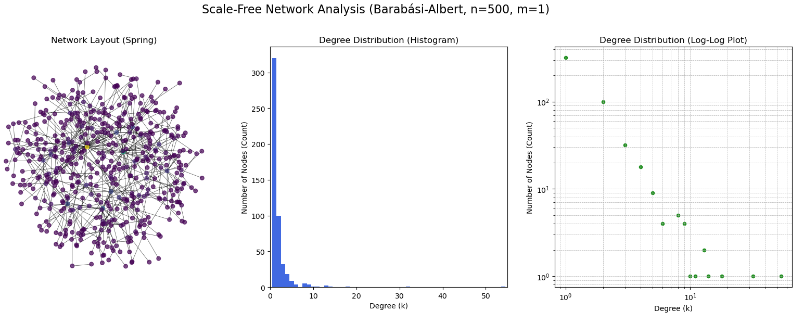


- Triangles: 3 – ABC, ABD, ADE
- Centered at A: $\binom{k_A}{2} = \binom{4}{2} = 6$
- Centered at B: $\binom{k_B}{2} = \binom{3}{2} = 3$
- Centered at C: $\binom{k_C}{2} = \binom{2}{2} = 1$
- Centered at D: $\binom{k_D}{2} = \binom{3}{2} = 3$
- Centered at E: $\binom{k_E}{2} = \binom{2}{2} = 1$
- **Transitivity:** $C_T = \frac{3 \times 3}{6+3+1+3+1} = 0.5$

“Scale-Free” Networks

- **Observation:** Network connections are not “democratic.”
- **Power-Law Degree Distribution:** Most nodes have very few connections, but a few nodes—the **“hubs”**—have a massive number of connections.
- **Examples of Hubs:**
  - **The Web:** Google
  - **Social:** An A-list celebrity
  - **Airlines:** Atlanta (ATL) or Dubai (DXB)

Example of scale-free network



Summary: How We Measure Networks

How do we quantify the “structure” we just discussed?

- **Degree Distribution:**
  - The count of connections for all nodes.
  - *Tells us:* Is it a “democratic” random network or a “hub-driven” scale-free network?
- **Average Path Length:**
  - The average distance (shortest path) between all pairs of nodes.
  - *Tells us:* How “small” is the world? How fast can something spread?
- **Clustering Coefficient:**
  - The average “cliquey-ness” of neighborhoods.
  - *Tells us:* Does the network form tight-knit groups?

Network models

Why Do We Need Network Models?

- **Create a baseline:** Models are a "null hypothesis." Is your real-world network just a random graph, or does it have special structure?
- **Understand properties:** They help us understand how properties emerge (e.g., how "six degrees of separation" can exist).
- **Simulate and predict:** We can simulate processes (like disease spread) on model networks.

We will cover three foundational models.

1. **Erdos-Renyi (Random)**
2. **Watts-Strogatz (Small-World)**
3. **Barabasi-Albert (Scale-Free)**

Erdos-Renyi (Random Graph)

- **The Idea:** The simplest "dumb" model.
- **How it's built (G(n, p)):**
  1. Start with `n` nodes.
  2. For *every possible pair* of nodes, add an edge between them with probability `p` **independently**.
- **Key Properties:**
  - Completely random, no "structure."
  - Degree Distribution (number of neighbors per node) follows a *Poisson* distribution.
  - Most nodes have about the same number of connections.
- **Real-world examples:** Not many. It's mostly a baseline for comparison.

Some interesting properties of ER random graphs $G(n, p)$

Degree distribution

- The degree distribution for any single node is exactly a binomial distribution: $B(n-1, p)$, so

$$P(\text{degree} = k) = \binom{n-1}{k} p^k (1-p)^{(n-1)-k}$$

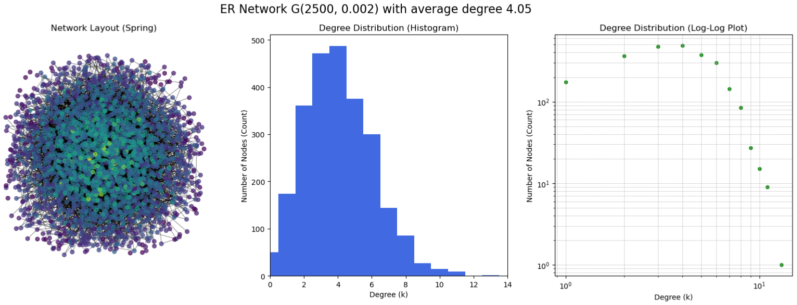
- the expected degree is $\lambda = (n-1)p$
- typically we deal with **large** networks, in this scenario:

$$\lim_{n\to\infty} \left[\binom{n-1}{k} p^k (1-p)^{(n-1)-k} \right] = \frac{\lambda^k e^{-\lambda}}{k!} \quad \text{where} \quad \lambda = (n-1)p$$

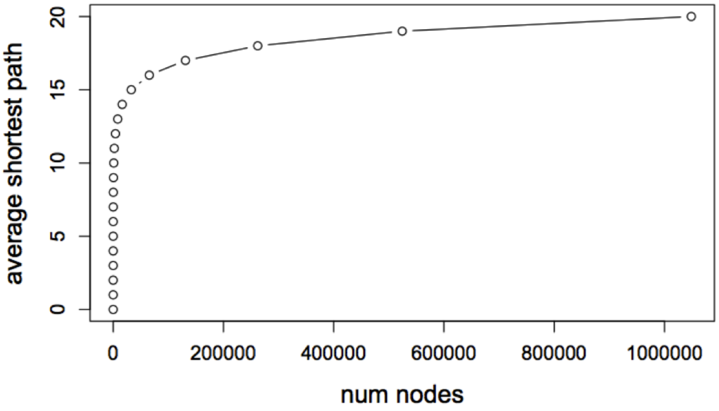
Erdos-Renyi example

```
import networkx as nx

G = nx.erdos_renyi_graph(2500, 0.002)
```



ER networks have short paths.



Diameter grows logarithmically for connected, sparse random networks

Suppose we have a $G(n, p)$ random network so that average degree is $\lambda \approx np$. Starting from an arbitrary node A, we count how many nodes we can reach after d hops

- at $d = 0$, reach 1 node
- at $d = 1$, reach λ nodes
- at $d = 2$, reach λ^2 nodes
- ...
- at $d = d$, reach λ^d nodes

so after $d \approx \frac{\ln(n)}{\ln(np)}$ hops, we reach whole network.

ER random networks do not show transitivity

Clustering coefficient is p. Which serves as a reference for the null model.

> ***In summary:*** *The classic ER model is good at explaining the "small-diameter" property but fails to explain the high clustering (cliques) and the existence of "hubs" seen in many real networks.*

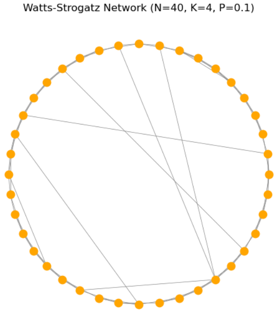
Watts-Strogatz (Small-World)

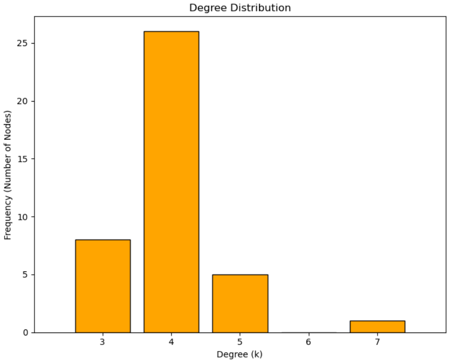
- **The Idea:** Explains the “six degrees of separation” phenomenon.
- **How it’s built:**
  1. Start with a regular ring lattice: `n` nodes in a circle, each connected to its `k` nearest neighbors. (High clustering, high path length).
  2. For every edge, with probability `p`, “rewire” it to a random node.
- **Key Properties (for small `p`):**
  - **High Clustering:** Your friends know each other (from the original lattice).
  - **Low Average Path Length:** You can get to anyone quickly (from the random “shortcut” links).
- **Real-world examples:** Social networks, food webs.

Watts-Strogatz example

```python
import networkx as nx

G = nx.watts_strogatz_graph(40, 4, 0.1, seed=42)
```





In summary: *The WS model successfully explains how networks can be highly clustered locally while being globally connected with short paths. It is the model for the "small-world" phenomenon.*

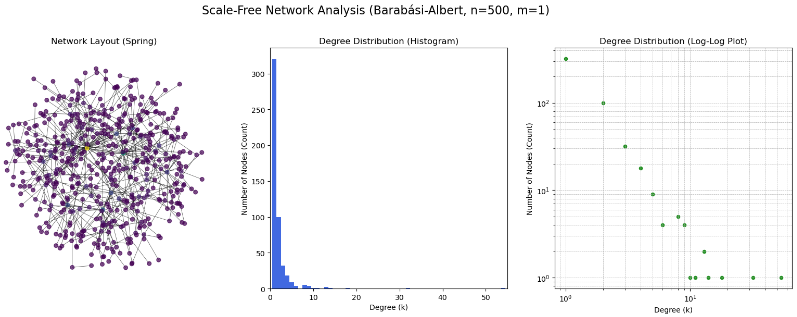
Barabasi-Albert (Scale-Free)

- **The Idea:** Models networks that *grow* and where popularity matters.
- **How it's built:**
  1. **Growth:** Start with a few nodes. Add one new node at a time.
  2. **Preferential Attachment:** When a new node joins, it connects to `m` existing nodes. The probability of connecting to a specific node is proportional to that node's current degree.
- **"The rich get richer":** New nodes prefer to link to nodes that are already popular (hubs).
- **Key Properties:**
  - **Scale-Free** Degree Distribution: A few "hub" nodes have a massive number of links, while most nodes have very few. Follows a Power Law.
- **Real-world examples:** The World Wide Web, citation networks, protein interaction networks.

Barabasi-Albert example

```python
import networkx as nx

G_ba = nx.barabasi_albert_graph(n=100, m=2)
```



In summary: *The BA model explains how "rich-get-richer" dynamics lead to the emergence of hubs and scale-free networks.*

Summary

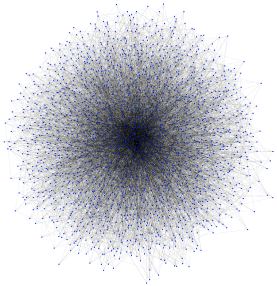
Property	Erdős-Rényi (ER) Model	Watts-Strogatz (WS) Model	Barabási-Albert (BA) Model
Degree Distribution	**Poisson** (Peaked at average)	**Peaked** (Like ER)	**Power Law** (Scale-Free)
"Hubs"?	No	No	**Yes (Definining feature)**
Average Path Length	**Short** ($O(\log N)$)	**Short** ($O(\log N)$)	**Very Short**
Clustering Coefficient	**Very Low**	**High (Definining feature)**	Low-to-Moderate
Real-World Example	(A simplified baseline)	Social networks, power grids	World Wide Web, citation networks

Node centrality

Identifying the “most important” nodes in a network.

Visual intuition often fails. Node importance metrics often help to:

- quantify **influence**: understand information flow and bridges
- identify **vulnerabilities**: susceptibility of attacks or critical points of failure, or super-spreaders
- ranking: pagerank!



Identifying the "most important" nodes in a network.

Classic Measures

- **Degree** Centrality: The "popularity" of a node. Simply the count of its connections (in-degree/out-degree for directed graphs).
- **Closeness** Centrality: The "efficiency" of a node. Measures the average shortest path from this node to all other nodes.
- **Betweenness** Centrality: The "brokerage"[2] of a node. Measures how often a node lies on the shortest path between two other nodes.
- **Pagerank** Centrality

[2] broker = intermediario, puente

Closeness centrality

The standard formula for closeness centrality is:

$$\text{closeness}(i) = \frac{1}{\sum_{j \neq i} d(i,j)}$$

This formula is only suitable for connected graphs. For graphs with more than one component, **harmonic centrality** is used.

Harmonic Centrality (for Disconnected Graphs)

$$\text{harmonic}(i) = \sum_{j \neq i} \frac{1}{d(i,j)}$$

In this formula, if node j is unreachable from i, the distance $d(i,j)$ is ∞, and its contribution $\frac{1}{\infty}$ is 0.

Normalization

To compare centrality scores across networks of different sizes, the scores are normalized dividing by the maximum possible score for nodes in graphs of the same size.

- **Normalized Closeness:**

$$\text{closeness}_{\text{norm}}(i) = \frac{n-1}{\sum_{j \neq i} d(i,j)}$$

- **Normalized Harmonic:**

$$\text{harmonic}_{\text{norm}}(i) = \frac{1}{n-1} \sum_{j \neq i} \frac{1}{d(i,j)}$$

Betweenness centrality

A high score indicates a node has significant influence over the network's flow. It is defined as:

$$\text{betweenness}(i) = \sum_{s \neq i \neq t} \frac{\sigma_{st}(i)}{\sigma_{st}}$$

Where:

- s and t are all possible pairs of nodes in the graph (other than i).
- σ_{st} is the total number of shortest paths between node s and node t.
- $\sigma_{st}(i)$ is the number of those shortest paths that pass *through* node i.

Normalization

To make scores comparable across graphs of different sizes, the raw score is divided by the **maximum possible score** a node could have in a graph with *n* nodes.

$$\text{betweenness}_{\text{norm}}(i) = \frac{\text{betweenness}(i)}{\binom{n-1}{2}}$$

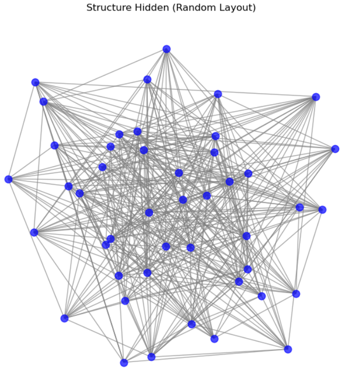
Centrality Measures & Applications: summary

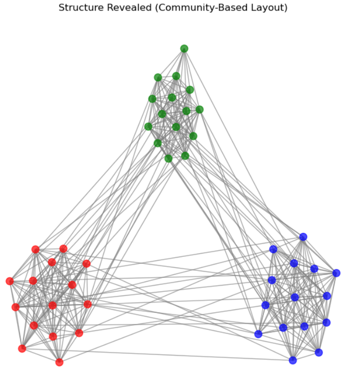
Measure	Role	Applications
Degree	**The Hub** (Popularity)	identifying influencers, airport logistics, malware sources
Closeness	**The Broadcaster** (Speed)	"patient zero" / epidemiology , emergency station placement , news dissemination
Betweenness	**The Bridge** (Control)	supply chain bottlenecks, critical infrastructure (power grids), information brokers
PageRank	**The Authority** (Influence)	Web search ranking, academic citations, ecological extinction cascades

Community Finding

Why look for communities?

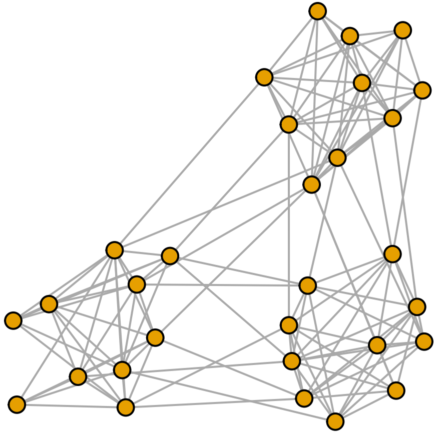
In complex systems, organization is rarely random. It is **modular**. Understanding its structure into sub-networks is key to “untangle” the mess.





What is a Community?

A set of nodes that are **densely connected internally** but **sparsely connected** to the rest of the network.



Defining communities

Let $C \subseteq V$ a candidate community of size $|C| = n_c$. Then

- maximize **internal density**: $\frac{m_c}{n_c(n_c-1)/2}$
- minimize **external density** or **cut ratio**: $\frac{f_c}{n_c(n-n_c)}$
- minimize **conductance**: fraction of edges leaving the cluster $\frac{f_c}{2m_c+f_c}$
- minimize **expansion**: nr of edges per node leaving the cluster $\frac{f_c}{n_c}$
- maximize **modularity**: $m_c - E_{random}[m_c]$ of a random graph[3]

where

- $f_c =$ nr. edges in the frontier of $C = |\{(u,v) | u \in C, v \notin C\}|$
- $m_c =$ nr. edges within cluster $C = |\{(u,v) | u, v \in C\}|$

[3] Here we use the *configuration model*: a random graph with same degree distribution as original network.

Community finding algorithms

There is a long list of algorithms, each optimizing their own metric, we will cover:

1. Girvan-Newman (hierarchical clustering) (Girvan and Newman 2002)
2. Louvain (modularity maximization)

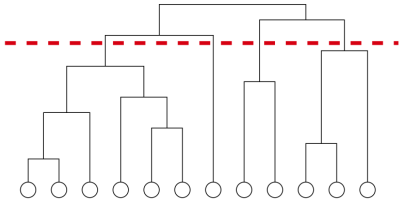
The **Girvan-Newman** (Girvan and Newman 2002) community finding algorithm

Edge betweenness centrality: number of shortest paths between all pairs of nodes in the network that pass through that specific edge.

Pseudocode

1. Compute betweenness for all edges in the network
2. Remove the edge with highest betweenness
3. Go to step 1 until no edges left

Result is a **dendogram**



Good results, prohibitive cost

The high complexity stems from the need to recalculate the edge betweenness centrality for all remaining edges every time an edge is removed.

In a sparse, unweighted graph, EBC for all edges can be calculated efficiently using a modified Breadth-First Search (BFS) approach (Brandes 2001)

The time complexity for a single full EBC calculation is $O(nm)$

There are m iterations at most, thus the time complexity is $O(nm^2)$ – if network is *sparse*, then $m \approx n$ and complexity is $O(n^3)$.

The Louvain Algorithm

Extremely fast and scalable, it is ideal for analyzing massive networks (e.g., social networks, web graphs).

- high-speed, hierarchical algorithm for detecting communities in large networks.
- **Goal:** to **maximize** a quality function called **Modularity (Q)**.
- It's a **greedy optimization** method that operates in two repeating phases.

What is Modularity (Q)? (Newman and Girvan 2004)

Modularity provides a single, principled score that tells us if our division of the network into communities is “good” or “meaningful.” Good $\approx$ **dense** inside, **sparse** outside

$Q =$ (Fraction of edges within communities)$-$(Expected fraction of such

- A high Q score (e.g., $Q > 0.3$) means the community structure is strong and surprising.
- A $Q \approx 0$ score means the partition is no better than random chance.

The entire definition hinges on one crucial question: **What does “at random” mean?**

The Null Model: What is “Random”?

We can’t just compare our network to a simple random (Erdős-Rényi) graph (degree distribution too different from real networks).

- A community with many hub nodes will *naturally* have many internal edges, just by virtue of its nodes’ high degrees.
- We need to distinguish structure that exists *beyond* what the degrees alone would predict.

Solution: We use the **Configuration Model** as our “random” null model.

The Configuration Model

1. **Stubs:** Take every node in the real network. For a node i with degree k_i, give it k_i “stubs” (half-edges).
2. **Total Stubs:** The total number of stubs in the entire graph is the sum of all degrees, which is equal to $2m$ (where m is the total number of edges).
3. **Random Wiring:** Create a random network by picking two stubs at random from the pool of $2m$ stubs and connecting them to form an edge.
4. **Repeat:** Repeat this process until all stubs are connected.

This random graph has the same nodes and degrees as our real graph, but all *structure* has been randomized.

Deriving the "Expected" Edges

Now we can use the Configuration Model to calculate the *expected* number of edges between any two nodes, i and j.

- Let k_i = degree of node i.
- Let k_j = degree of node j.
- Let $2n$ = total number of stubs (sum of all degrees).

The probability of a single stub from node i connecting to any stub from node j is:

$$P(\text{stub } i \to \text{stub } j) = \frac{k_j}{2m}$$

(It's k_j because j has k_j stubs, out of a total of $2m$ stubs in the network).
Since node i has k_i stubs to connect, the *expected* number of edges between i and j is:

$$E[A_{ij}] = k_i \times P(\text{stub } i \to \text{stub } j) = k_i \left(\frac{k_j}{2m}\right) = \frac{k_i k_j}{2m}$$

The Full Modularity Formula (Sum over Nodes)

Modularity Q is the sum of the "surprise" for every pair of nodes, normalized by the total number of edges.

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

Where $\delta(c_i, c_j)$ is a *community filter*: it is 1 if i and j are in the *same* community ($c_i = c_j$), and 0 otherwise.
In plain English: "For every pair of nodes *in the same community*, sum up the difference between the *actual* edges and the *expected* edges."

Alternative Formula (Sum over Communities)

The previous formula is often re-written as a sum over communities c, which can be more intuitive.

$$Q = \sum_{c} \left[\frac{m_c}{m} - \left(\frac{K_c}{2m} \right)^2 \right]$$

where $K_c =$ sum of degrees of all nodes *in* community c.
This is mathematically equivalent to the first formula and is often easier to compute.

Interpreting Q:

- $Q > 0.3$: Generally considered a strong, significant community structure.
- $Q \approx 0$: The partition is no better than random.
- $Q < 0$: Fewer internal links than expected (e.g., a disassortative or bipartite-like structure).

- **NP-Hard:** Finding the *true* maximum Q score is computationally infeasible (NP-hard). Algorithms like Louvain and Leiden find very good "local" maxima using greedy optimization.

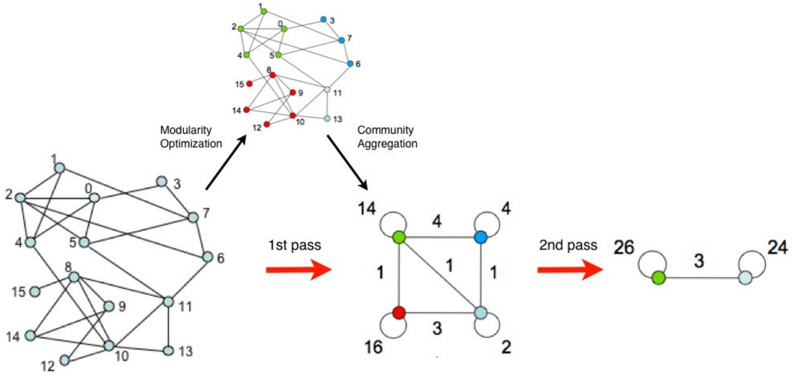
The Louvain Algorithm: A Two-Phase Process

The Louvain algorithm iterates through two phases until no further improvement in Modularity (Q) can be made.

1. **Phase 1: Modularity Optimization (The “Move”)**
   - Nodes are locally optimized by moving them between communities.
2. **Phase 2: Community Aggregation (The “Collapse”)**
   - The identified communities are collapsed into new “super-nodes” to build a new, smaller network.

This entire process is then repeated on the new, smaller network.

The Algorithm: A Two-Phase Process, cont.



Phase 1: Modularity Optimization

This phase involves locally optimizing the modularity by moving individual nodes.

1. **Initialization:** Place every node in its own unique community (e.g., n nodes $= n$ communities).
2. **Node Iteration:** For each node i:
   - Consider all its neighbors j.
   - Calculate the **change in modularity (ΔQ)** that would occur if node i were moved from its current community into the community of neighbor j.
   - Move node i to the neighboring community that results in the **maximum positive** ΔQ. (If all moves result in a negative ΔQ, the node stays put).
3. **Repeat:** This process (Step 2) is repeated for all nodes until no single node move can improve the overall modularity Q. A local maximum has been reached.

Phase 2: Community Aggregation

This phase creates a new, smaller, *weighted* graph based on the communities found in Phase 1.

1. **Collapse Communities:** All nodes that were assigned to the *same* community in Phase 1 are “collapsed” into a single new **super-node**.
2. **Create New Edges:** Edges between the new super-nodes are created:
   - The weight of the edge *between* two super-nodes is the **sum of all edge weights** between the original nodes in those two communities.
   - The weight of a **self-loop** on a super-node is the **sum of all internal edge weights** of the original community.
3. **Result:** A new, coarsened graph where the super-nodes represent the communities from the previous level.

Iteration and Hierarchy

These two phases form **one "pass"** of the algorithm.

- The algorithm then **repeats**, applying **Phase 1** and **Phase 2** to the new, coarsened graph from the previous pass.
- This continues until the algorithm converges—meaning a pass completes with **no changes** in modularity and only one (or a few) super-nodes remain.

Result: A Hierarchy of Communities

Summary

Pros:

- **Fast & Scalable:** One of the fastest methods available, with a complexity often near $O(m \log n)$ or $O(n \log n)$ for sparse graphs.
- **Hierarchical:** The multi-pass process naturally reveals a hierarchical community structure.
- **Intuitive:** The greedy optimization process is straightforward to understand.

Cons:

- **Greedy:** It is not guaranteed to find the *global* optimum for Q (it finds a local maximum).
- **Resolution Limit:** Like all modularity-based methods, it may fail to find very small communities.
- **Key Flaw (solved by Leiden (Traag, Waltman, and Van Eck 2019)):** Can sometimes produce "ill-defined" communities (e.g., loosely connected or even disconnected subgraphs) that are still joined into one community.

References

Backstrom, Lars, Paolo Boldi, Marco Rosa, Johan Ugander, and Sebastiano Vigna. 2012. "Four Degrees of Separation." In *Proceedings of the 3rd Annual ACM Web Science Conference*, 33–42. WebSci '12. Evanston, IL, USA: ACM. https://doi.org/10.1145/2380718.2380723.

Brandes, Ulrik. 2001. "A Faster Algorithm for Betweenness Centrality." *The Journal of Mathematical Sociology* 25: 163–77. https://doi.org/10.1080/0022250X.2001.9990249.

Faloutsos, Michalis, Petros Faloutsos, and Christos Faloutsos. 1999. "On Power-Law Relationships of the Internet Topology." In *Proceedings of the ACM SIGCOMM '99 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 251–62. New York, NY, USA: ACM. https://doi.org/10.1145/316188.316229.

Girvan, Michelle, and Mark E. J. Newman. 2002. "Community structure in social and biological networks." *Proceedings of the National Academy of Sciences* 99 (12): 7821–26.

Leskovec, Jure, and Eric Horvitz. 2008. "Planetary-Scale Views on an Instant-Messaging Network." In *Proceedings of the 17th International Conference on World Wide Web (WWW '08)*, 915–24. New York, NY, USA: ACM. https://doi.org/10.1145/1367497.1367620.

Milgram, Stanley. 1967. "The Small World Problem." *Psychology Today* 1 (1): 60–67.

Newman, Mark E. J., and Michelle Girvan. 2004. "Finding and evaluating community structure in networks." *Physical Review E* 69 (2): 026113. https://doi.org/10.1103/PhysRevE.69.026113.

Traag, Vincent A, Ludo Waltman, and Nees Jan Van Eck. 2019. "From Louvain to Leiden: guaranteeing well-connected communities." *Scientific Reports* 9 (1): 5233. https://doi.org/10.1038/s41598-019-41695-z.

Watts, Duncan J., and Steven H. Strogatz. 1998. "Collective Dynamics of 'Small-World' Networks." *Nature* 393 (6684): 440–42. https://doi.org/10.1038/30918.