Graph Analysis & Random Graphs

Scribe: Elliot Pickens (eep58)

Anna Scaglione
Graph-Based Data Science for Networked Systems

Feb. 18, 2022

Contents

Lecture 7

1 Introduction

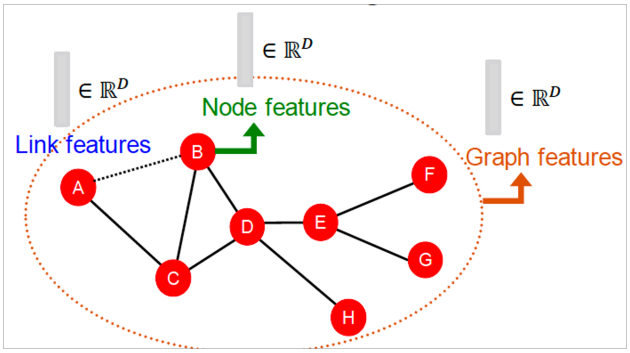
Previously, we defined a number of basic graph theory concepts for weighted and unweighted graphs ($\mathcal{G} = (\mathcal{V}, \mathcal{E})$) using a number of objects including:

- Adjacency matrices $\mathbf{A}$
  - To represent the basic connectivity structure of a graph's nodes
- Incidence matrices $\mathbf{B}$
  - To represent the basic structure while differentiating nodal quantities and model the impact of edge flows
- Laplacian matrices $\mathbf{L}$
  - To represent a graph spectrum derived from a combination of $\mathbf{A}$ & $\mathbf{B}$ since $\mathbf{L} = diag(\mathbf{d}) - \mathbf{A} = \mathbf{B}\mathbf{B}^T$

Today, we'll look some other metrics and measures we can use to understand the structure of a network.

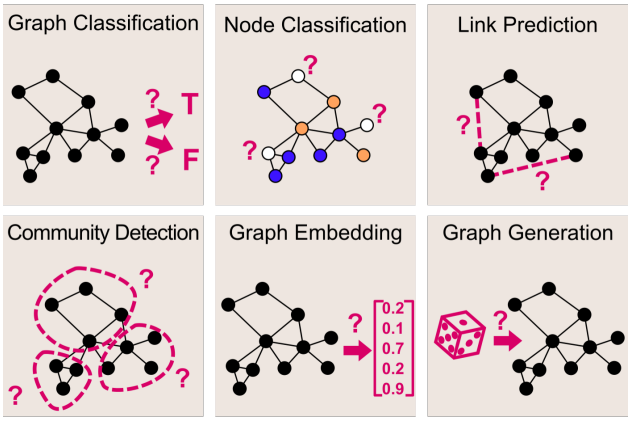
2 Graph Analysis

2.1 Premise



One aspect of complex networked systems lies in the analysis of the network itself i.e. the (weighted) graph. Through this analysis we can carry out inference tasks on the elements of the graph. These elements include:

- Individual nodes
- Individual links
- The entire graph



Using these basic components we can carry out a number of machine learning tasks like:

- Graph classification
- Node classification
- Link prediction
- Community detection
- Graph embedding
- Graph generation

2.1.1 Node Inference

These tasks apply to a broad number of real world scenarios. Focusing first on **nodes**, we can use node inference to understand networks by exploiting **network homophily** or **assortativity**.

Definition 2.1 (**Network Homophily or Assortativity**)**.** Networks represent, or cause a tendency for nodes to share attributes with their neighbors in the graph. This is what makes the structure of the network useful for the inference task.

We can apply this to systems like:

- **Social Networks:** to identify bots in online social networks, or radicalized terrorists
- **Biological Networks:** to classify the function of proteins in the interactome
- **Knowledge Networks:** to classify the topic of a document within a citation network
- **The internet:** to classify the nodes generated by a Sybil attack (fake nodes in a peer to peer network)

2.1.2 Relation Prediction or Network Inference

We can also work with a system's edges by trying to *unveil* them in a useful manner. A few of the many application that use this approach are:

- **Social Networks:** helping design recommender systems for social media platforms (e.g. friend recommendations)
- **Biological Networks:** predicting biological outcomes/interactions (e.g. discovering drug side effects)
- **The internet:** uncovering source and destination pairs within internet traffic

- **Grids:** generating statistically consistent sample grids and filling up data gaps. This can help with the understanding of certain sections of a network

Relation prediction is also directly related to another topic that we already discussed in class: *clustering* graphs into *communities*.

2.1.3 Graph Classification and Regression

If we have sufficient graph data then we can perform both **regression** and **classification** on said data. That is to say that we can use previous examples of other graphs to fit models that can help us classify, match, or cluster graphs.

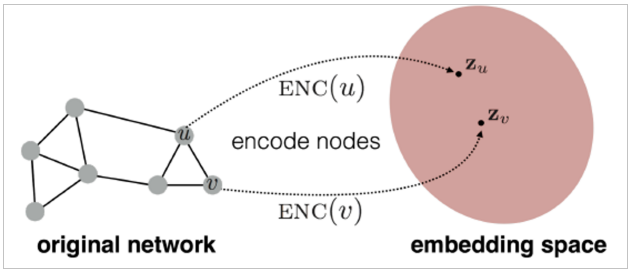
We can also create **graph embeddings** to aid us in inference tasks (supervised or unsupervised) by creating useful alternative graph representations. Graph embeddings can take a number of different forms, but one recently popularized format are graph embeddings that take inspiration from natural language processing to create vector representations of graphs.[1]

If we can capture the properties of a graph or set of graph ensembles using a distribution then we can potentially do **graph generation** and Bayesian inference. This can mean using graph data to generate a Bayesian network, which we can use as needed.

2.1.4 Feature Extraction

For the classification and regression tasks mentioned above to be done effectively we need to have solid data. This can be difficult when working with graphs given the representations we have been working with up to this point (**A**, **B**, & **L**) can be tricky to work with for these tasks. To get around this we perform feature extraction where we find **features** by finding functions that can map our graphs onto representative data points. Those functions can map elements of $\mathcal{V}$, $\mathcal{E}$, or the entire graph $\mathcal{G}$ onto a point $\mathbb{R}^d$. We hope that d is small, but ultimately what matters is that it is large enough to be informative.

2.1.5 Node Embeddings

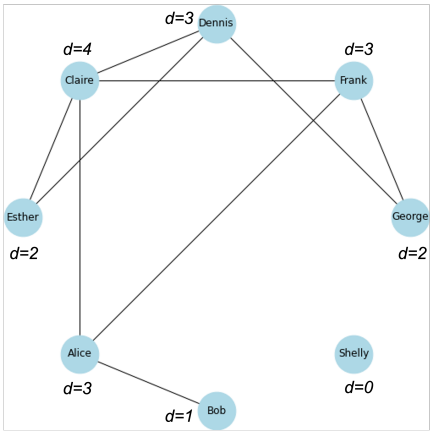


Among the various types of graph embeddings, one frequently used type are **node embeddings**. *Shallow* node embeddings generate a matrix $Z \in \mathbb{R}^{d \times |\mathcal{V}|}$ whose columns are the feature vectors of each node $z_v \forall v \in \mathcal{V}$. Previously I hinted at deep learning based embedding methods, but in this set of notes we will focus on classical nodal feature components of z_u like node degree, different forms of node centrality, clustering coefficients, and others.

[1] Check out this article for a good summary of some deep learning based graph embedding methods: Graph Embeddings - The Summary

2.2 The Ranking of Nodes: Centrality Measures

2.2.1 Nodal Degree



Starting from the basics, the first feature we can define is **nodal degree**. We already defined a node's degree $d_v = |\mathcal{N}_u|$ as the number of neighbors of a node. As well as the degree sequence as a feature of the graph. We can use this as a 1D feature.

For a directed graphs we have in degree and out degrees d_v^+ and d_v^-, which provide a 2D feature component for each node.

Note: In this section of the notes (2.2) we will focus on undirected graphs unless otherwise specified.

2.2.2 Degree Distribution

We can also use the empirical distribution of the nodal degree as a feature vector. We generate such a vector as the fraction of nodes that have a given nodal degree. Put formally we can say,

$$p_k = \hat{P}(d_v = k) = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} \delta(d_v - k), \quad k = 1, \ldots, \overline{d} \tag{1}$$

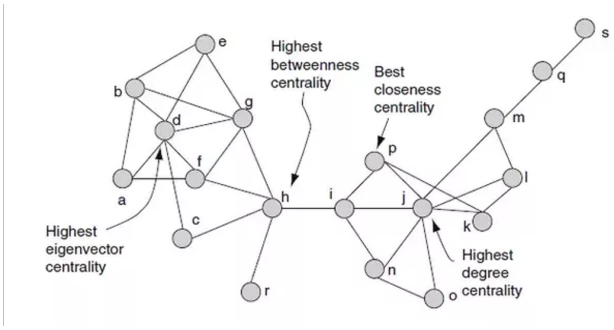
where $\overline{d} = \max_{u \in \mathcal{V}}(d_u)$ and $\delta(k)$ is the Kronecker delta, i.e the indicator function defined as:

$$\delta(k) := \begin{cases} 1 & \text{for } k = 0 \\ 0 & \text{else} \end{cases}$$

We can use 1 to define an alternative definition of the average degree:

$$d_{\text{av}} = \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} d_v = \sum_{k=1}^{\overline{d}} k p_k \tag{2}$$

2.2.3 Node Centrality Measures



Definition 2.2 (**Centrality**)**.** Centrality is a measure used to establish the **most important node in a network**. The definition of *importance* can vary based on the specific measure of centrality.

2.2.4 Degree Centrality

Degree centrality is one measure of centrality. In degree centrality we define the central node to be the node with the maximum degree.

$$v^{\text{deg}} = \underset{v}{\text{argmax}}\, d_v. \tag{3}$$

This is a fairly simple centrality measure, but it can be very useful as the node with the greatest degree often happens to be quite important. Such a node can represent a very important person within a social network, or a point of potential congestion within a infrastructure network.

2.2.5 Eigenvector Centrality

Eigenvector centrality is an alternative centrality measure that captures the fact that **central nodes** must have **central neighbors** through the recurrence relation:

$$z_u \propto \sum_{v \in \mathcal{N}_u} z_v \quad u \in \mathcal{V} \quad \Rightarrow \quad \lambda z = Az \tag{4}$$

So which eigenvector do we use? We know there are N eigenvectors and that the score is proportional to λ. We also know that A has non negative entries and **Perron Frobenious Theorem** guarantees that the **maximum eigenvector has positive entries.** Thus, the eigenvector centrality of a node a node is the corresponding entry of the leading eigenvector (which will assign a positive score to every node).

Let z_1 be the leading eigenvector s.t. $\lambda_1(A)z_1 = Az_1$. Then the most central node is[2]:

$$v^{\text{eig}} = \underset{v}{\text{argmax}}[z_1]_u \tag{5}$$

Note: in disconnected networks (where there multiple components) it is useful and more meaningful to look for the central node in each component.

In figure 2.2.5 above we can see how degree centrality and eigenvector centrality vary.

[2]Any scaling of the leading eigenvector will give the same result.

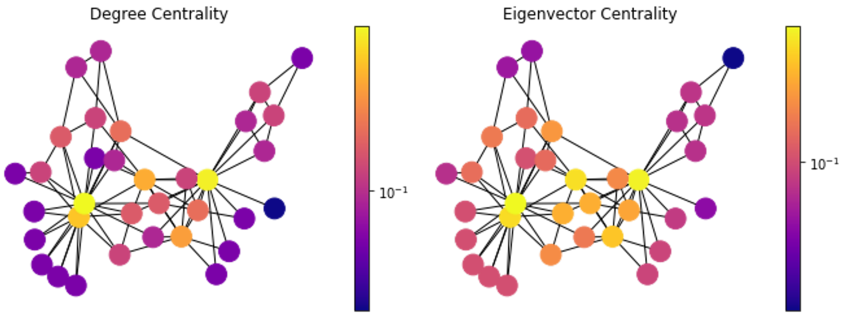


Figure 1: Zachary's Karate Club Centrality

2.2.6 Eigenvector Centrality for Directed Networks

We also need to understand centrality within directed graphs. A few examples of why such a measure can be useful are:

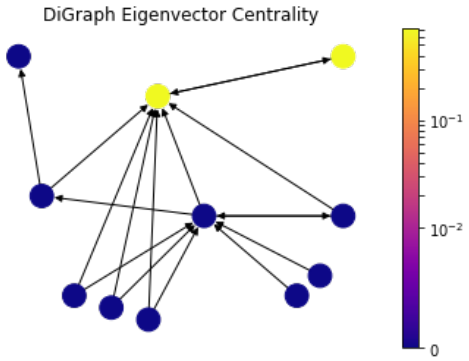
- The notion of centrality for directed networks is very important for knowledge networks
- It is used in **webometrics**, i.e. the part of network science that uses network embeddings to rank and organize web-pages on the web
- It is useful for citations networks organizing the papers based on how authoritative they are, as well as the hub that connects important documents

In directed networks $\mathbf{A}$ is not symmetric, so its left and right eigenvectors are not equivalent. This makes applying the logic we used in 2.2.5 problematic. To get around this we need to choose the right set of eigenvectors, which is this case in the set of right eigenvectors since ***centrality is bestowed by the nodes that point towards you***. We can also apply the same logic to degree centrality for directed graphs via **in degree centrality**.

Therefore, we can use the right eigenvectors in $\lambda_1(A)z_1 = Az_1$ to continue to use the definition of eigenvector centrality laid out in 5.

Caveat: The definition laid out above can be problematic because:

- if a node has zero in degree has zero centrality score, which may be ok;
- the problem is that if a node has a large in-degree, but all its in-neighbors have zero in-degree then that node will also have a score of zero



This can take place in a directed acyclic graph like a article citation network, where all nodes get null eigenvector centrality. Only vertices that are in a strongly connected component of two or more vertices, or the out-component of such a component, can have non-zero eigenvector centrality. To account for this we need other definition, particularly for knowledge networks.

2.2.7 Katz Centrality

One solution is to add some *centrality* β to all nodes, equally:

$$z_v = \alpha \sum_{u \in \mathcal{N}_v} z_u + \beta, \quad \Rightarrow \quad z = \alpha A + \beta 1 \tag{6}$$

where now α is a parameter. The centrality score is now:

$$z^{\text{Katz}} = \beta(I - \alpha A)^{-1} 1. \tag{7}$$

but we need to choose α and β. β is a sort of "*self-estime*" constant, but scales everything equally and does not help the ranking.

Note: that a variation of Katz centrality chooses a vector β with different components based on some side information on the importance of that node.

- At $\alpha = 0$ the measure is meaningless $z = \beta 1$
- The values of α that make the matrix not invertible, i.e. such that $\det(I - \alpha A) = 0$, are the eigenvalues of A
- The one solution that would weight A the most, while $\det(I - \alpha_1 A) \approx 0$.
- $\det(I - \alpha_1 A)$ is the characteristic equation, again, $\lambda_1(A)$
- Values $\alpha \approx \lambda_1(A)$ are the popular choice:
  - For undirected or non-acyclic graphs this choice gives results close to the eigenvector centrality
  - For directed networks it does give some weight as well

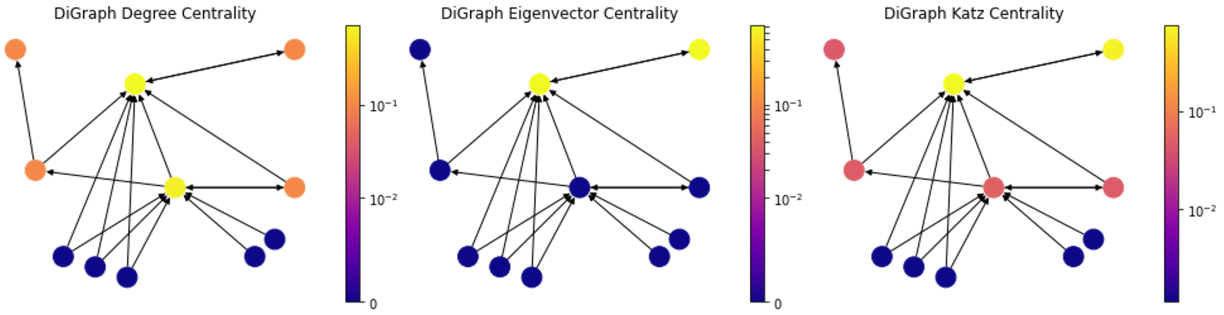


Figure 2: Centrality Measure Comparison

The plots in 2.2.7 show the differences between the centrality measures.

The main drawback to Katz centrality is that a nodes with high centrality spread its high centrality to its out neighbors. When such a node has many out edges, it ends up pushing on that centrality to many other nodes. For example, Amazon and eBay point to tons of vendors sites, but are they important? A variant of Katz centrality that makes sure this does not happen is **PageRank** centrality.

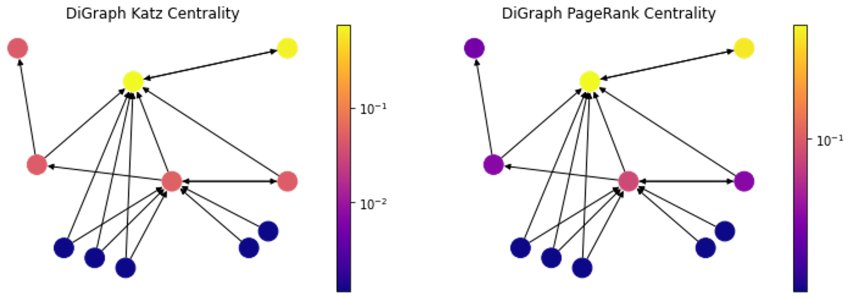
2.2.8 PageRank Centrality

With PageRank centrality, we divide the adjacency matrix by the out degree. To do this we first recognize that the out-degree sequence is $d^{out} = A^\top 1$, because $d_v^{out} = |\mathcal{N}_v^-| = \sum_{u \in \mathcal{V}} a_{uv}$. This does, however, include the case where $d_v^{out} = 0$, which we can fix by defining $[d^{+out}]_v = \max(d_v^{out}, 1)$ We then normalize the matrix A by scaling every column by $a_{u,v}/d^{+out}$.

$$z = \alpha A \text{diag}^{-1}(d^+)x + \beta I, \tag{8}$$

$$\Rightarrow \quad z^{\text{PageRank}} = \beta(I - \alpha A \text{diag}^{-1}(d^+))^{-1}1 \tag{9}$$

Just like with the Katz centrality, in relative terms β does not matter, so we can just choose $\beta = 1$.



In the figure above we can see the different between these two measures.

2.2.9 Some Observations

So far, the forms of centrality we have considered highlight well connected nodes. These nodes are either at the receiving end of their neighborhood, or are situated as authorities within the network.

Another way measure centrality is by looking at the nodes needed to reach others. We call these nodes hubs, because they play an important role in network traversal. One example of a hub is a web page that is mostly an index that links to other pages.

The forms of centrality we define next focus on highlighting this notion of centrality and identifying hubs.

2.2.10 Authority and Hub Centrality

Our goal in this section is to discover both hubs and authorities. One way we can do this is to find a way to simultaneously define both hubs an authorities. To do this, let x and y be the authorities and hub centralities and assert that

- **Authority centrality**: The authority score is proportional to the hub centralities it connects to
- **Hub centrality**: The hub score is proportional to the authority centralities it connects to

Mathematically that means:

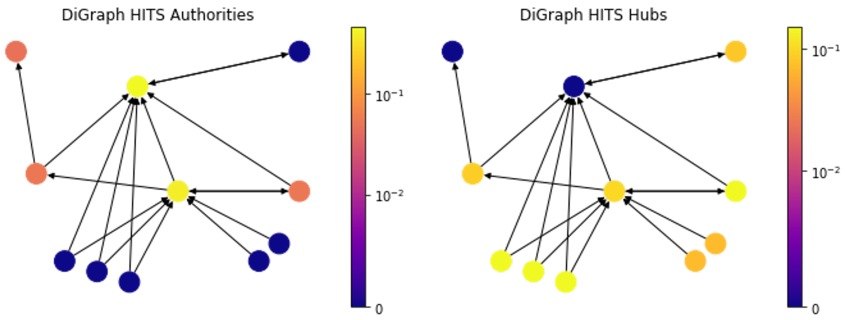
$$x = \alpha A y, \quad y = \beta A^\top x \tag{10}$$

Combining the two equations we obtain the eigenvectors equations:

$$\lambda x = A A^\top x, \quad \lambda y = A^\top A y. \tag{11}$$

The eigenvalues of $AA^\top$ and $A^\top A$ are the same. Their eigenvectors are the left and right eigenvectors respectively. In fact, these are the **singular vectors**, and the equations that yield the **singular value decomposition** of a matrix:

$$A = X\Sigma Y^\top \quad \text{where} \quad \sigma_n = \sqrt{\lambda_n} \tag{12}$$



This pair of centrality measures (x_v, y_v) for each node $v \in \mathcal{V}$ has been defined for hyperlink networks, and its calculation is called *hyperlink-induced topic search (HITS)*. HITS tackles the problem with zero in-degree nodes of Eigenvector centrality by turning zero in-degree nodes into central hubs that contribute to other nodes. We cam see an example of HITS in 2.2.10.

2.2.11 Closeness Centrality

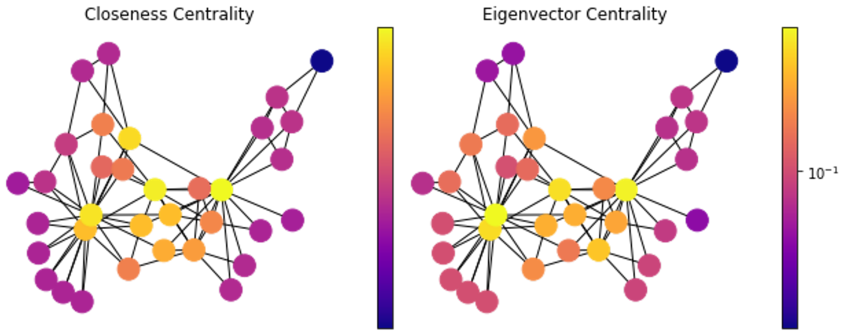
Now we are going to consider two forms of centrality that are based on **distance** (the shortest paths within networks). To start off we will take a look at closeness centrality.

Closeness centrality measures depends on the average shortest path from node v to every other node in the network. To define this, let $d(v, u)$ be the shortest path between node v and the node u. Then

$$\text{Average shortest path}: \quad \ell_v = \frac{1}{N}\sum_{u\in\mathcal{V}} d(v, u) \tag{13}$$

We would use this as a centrality measure, because it would rank high when far from everyone. Thus, we can define **closeness centrality** as the reciprocal of the average shortest path:

$$z_v^{\text{close}} = \frac{1}{\ell_v} \tag{14}$$



Suppose an actor (Christopher Lee) was in many (unimportant) movies, where few other stars were cast. Then if you construct a graph that connects actors on the same movie, Christopher Lee has small degree and

eigenvector centrality - but also has a high closeness centrality.

For **Disconnected networks** where there is more than one component, the distance between nodes can be infinite (if they are in separate components). To get around this there are two options:

1. Measure only distances in a certain component.

   - **Problem**: If the components have very different sizes, nodes in small components will have greater closeness centrality

2. Revise the definition, as follows:

$$z_v^{\text{close-2}} = \frac{1}{N-1} \sum_{u \neq v} \frac{1}{d(v,u)}$$

   - This definition solves our problem, because if v and u are in different components then node u has no contribution to the centrality metric:

$$d(v,u) = \infty \quad \Rightarrow \quad \frac{1}{d(v,u)} = 0$$

2.2.12 Betweenness Centrality

Betweenness centrality measures how frequently a node is in a path between other nodes. These nodes are crucial to flow of information, particularly in clustered networks.

To dig into the measure a little, consider an undirected network where the shortest path between any node pair is unique. The betweeness centrality is the number of times node v appears in the shortest path connecting the network pairs.

$$z_v^{\text{betw}} = \sum_{(s,t) \in \mathcal{V} \times \mathcal{V}} n_{s,t}^v \tag{15}$$

where

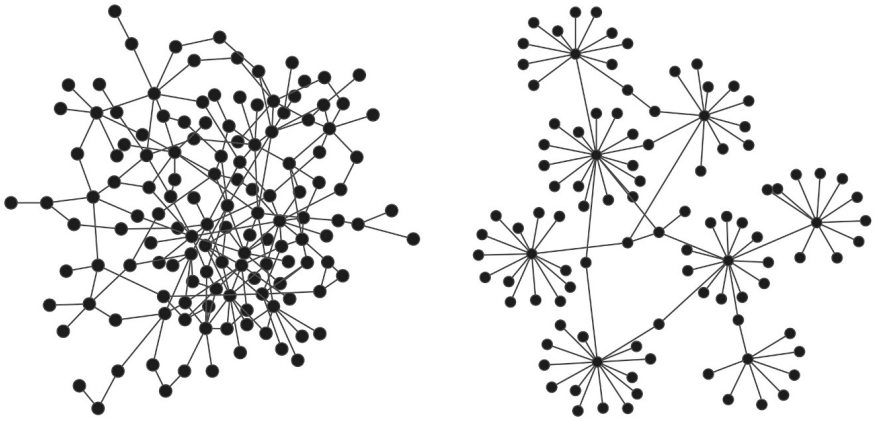
$$n_{s,t}^v = \begin{cases} 1 & v \text{ is in the path between } (s,t) \\ 0 & \text{else} \end{cases} \tag{16}$$

This metric works for both directed and undirected graphs.

3 Groups of Nodes

3.1 Homophily

In addition to ranking the nodes there are other metrics that can be useful for understanding trends within a network. **Homophily** is a metric that tries to assess how **homogeneous** the neighborhoods of neighbors are with respect to a certain feature. In the next section we will focus on their degrees, but other features can be chosen instead.



(A) Assortative (B) Disassortative

In an **assortative network** any node u has neighboring nodes that are similar with respect to the feature $\phi(u) \approx \phi(v), v \in \mathcal{N}_u$. A disassortative network showcases the opposite trend.

3.2 Measure of Assortativity

For **degree assortativity** $\phi(u) = d_u$, which is showcased by the networks in the previous section. The **Pearson correlation coefficient** $r \in [-1, 1]$ of **degrees of neighboring nodes** is one another measures that can be used:

$$r = \frac{\sum_{(u,v)\in\mathcal{E}}(\phi(u) - \overline{\phi}_{\text{in}})(\phi(v) - \overline{\phi}_{\text{out}})}{\sqrt{\sum_{(u,v)\in\mathcal{E}}(\phi(u) - \overline{\phi}_{\text{in}})^2}\sqrt{\sum_{(u,v)\in\mathcal{E}}(\phi(v) - \overline{\phi}_{\text{out}})^2}} \tag{17}$$

$$\overline{\phi}_{\text{in}} = \frac{\sum_{(u,v)\in\mathcal{E}}\phi(u)}{|\mathcal{E}|}, \quad \overline{\phi}_{\text{out}} = \frac{\sum_{(u,v)\in\mathcal{E}}\phi(v)}{|\mathcal{E}|} \tag{18}$$

If $r > 0$ the network is assortative if $r < 0$ the opposite is true.

3.3 Cosine Similarity

Cosine similarity is another measure of similarity for undirected networks. Cosine similarity works by comparing the common neighbors of neighborhoods to find their similarity. To do this let's calculate the number of common neighbors:

$$|\mathcal{N}_u \cap \mathcal{N}_v| = n_{u,v} = \sum_{k\in\mathcal{V}} a_{uk}a_{kv} = [A^2]_{u,v} \tag{19}$$

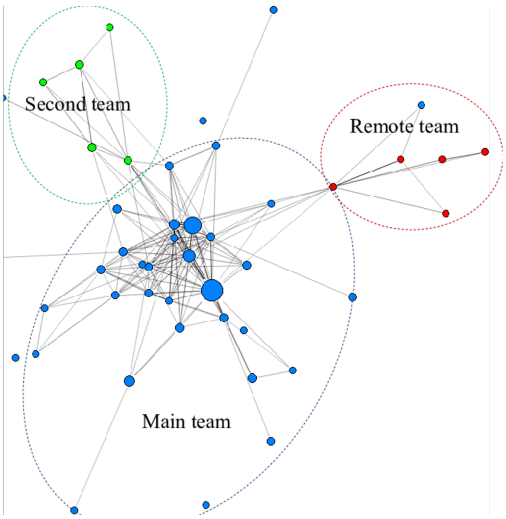
Since $A = A^\top$, $A^2 = A^\top A$ then $n_{u,v} = [A^2]_{u,v} = [A^\top A]_{u,v}$. In words $n_{u,v}$ is the projection of the columns of A corresponding to nodes u and v.

For two vectors:

$$\cos(\theta_{xy}) = \frac{x^\top y}{\|x\|\|y\|}, \tag{20}$$

which suggests that we can use a normalized version of $n_{u,v}$:

$$\sigma^{\cos}_{u,v} = \frac{[A^2]_{uv}}{\sqrt{[A^2]_{uu}}\sqrt{[A^2]_{vv}}} \tag{21}$$



Cosine similarity (and other similarity measures) are often useful for identifying nodes, or group of nodes, with different roles or types of interactions. The example in figure 3.3 shows how this can work for a social network.

3.4 Jaccard Coefficient

Another measure of similarity is the **Jaccard coefficient**,

$$\sigma_{uv}^{\text{Jaccard}} = \frac{n_{u,v}}{d_u + d_v - n_{u,v}} \tag{22}$$

Here $d_u + d_v$ is the number of neighbors of u plus those of v, which we can write using set notation as $d_u + d_v = |\mathcal{N}_u \cup \mathcal{N}_v|$. Thus, the Jaccard coefficient measures how much the two neighborhoods overlap:

- the higher the overlap $\rightarrow$ the greater the value of $\sigma_{uv}^{\text{Jaccard}}$,
- if u and v have no common neighbors then $\sigma_{uv}^{\text{Jaccard}} = 0$

3.5 Special Subgraphs: Cliques and k-cores

We can also identify special groups in a network by finding tightly connected sub-graphs. This is particularly useful when they are enmeshed in sparse graphs.

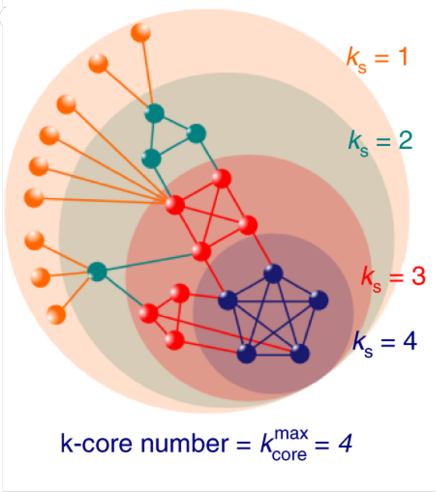
3.5.1 Cliques and k-cores

Definition 3.1 (Clique)**.** A clique is a **complete subgraph**

A less restrictive definition for a group of nodes is that of k**-core**, which is a **sub-set of nodes where each of them is connected to at least** k **others in the set**.

As we can see in figure 3.5.1, for all k-cores the following hold:

- There is no k core for $k > \max_u(d_u)$
- One can show that for $k > k'$ the k-cores are subsets of the k'-cores



3.6 Transitivity

Transitivity applies to a mathematical relation among the elements of a set. Examples of transitivity appear everywhere including algebra. Equalities and inequalities, for example, are transitive as $a = b$ and $b = c$ implies $a = b$, etc.

For nodes in a graph, this can be applied to predicates regarding pairs of nodes, like the following one based on connectivity:

> If there is an edge between node u and node v and one between node v and node w, then there is an edge between node u and w.

In words, the nodes u, v and w form a cycle of length 3.

3.7 Clustering Coefficients

Having perfect transitivity is meaningless, because that would mean that every component is a clique, but it is interesting to measure how many nodes (on average) form the sorts of triplets described in 3.6.

This might be useful when analyzing something like a social networks where this would measure how more likely it is that a *friend of a friend is a friend*. In fact, empirical observations show this is often the case. Collaborators networks, for instance, have clustering coefficients in the order of 0.2.

- This is high for large networks: if friends where chosen at random, the clustering coefficient should be $O(1/N)$

Getting into the math, the clustering coefficient is defined as follows:

$$C = \frac{\text{(number of closed paths of length two)}}{\text{(number of paths of length two)}} \tag{23}$$

or equivalently, considering that each triangle contributes 6 paths of length 2, it is:

$$C = \frac{\text{(number of triangles)} \times 6}{\text{(number of paths of length two)}} \tag{24}$$

3.7.1 Local Clustering

We can also develop an associated a nodal metric that characterizes how clustered the neighborhood of a node is as

$$C_v = \frac{\text{(number neighbors in } \mathcal{N}_v \text{ that are connected)}}{\text{(number of pairs of neighbors)}} \tag{25}$$

where the number of pairs of neighbors $= |\mathcal{N}_v|^2$.

The value of C_v is often found to correlate with d_v. And Watts and Strogatz showed that this can be used as an alternative definition of clustering coefficient:

$$C^{\text{WS}} = \frac{1}{N} \sum_{v \in \mathcal{V}} C_v \tag{26}$$

3.7.2 Reciprocity

The clustering coefficient measures the frequency of loops of length three in a network, but why are we only concentrating on loops of length three? Sometimes looking at two way links can be more useful. Such bidirectional relationships are particularly interesting in directed networks. We can define such cases by stating that if there is a directed edge from node u to node v and one from v to u then the edges between u and v are **reciprocated** or are **co-links**.

The graph **Reciprocity** r^{rec} is defined as the fraction of edges that are reciprocated. And noting that the product of adjacency matrix elements $a_{uv}a_{vu} = 1$ if and only if the edges are reciprocated, the graph reciprocity is:

$$r^{\text{rec}} = \frac{1}{|\mathcal{E}|} \sum_{uv} a_{uv}a_{vu} = \frac{1}{|\mathcal{E}|} \text{Tr}(A^2) \tag{27}$$

where the average is calculated with respect to the total number of (directed) edges in the network.

4 Conclusion

In this section:

- We discussed how to extract features form graphs, based on their structure
- We defined local (nodal) and group properties:
    1. Degree distribution
    2. Centrality metrics
    3. Similarity metrics and assortatitive networks
    4. Clustering coefficient
    5. Reciprocity

In the next section we will look at models for Random Graphs.

Lecture 8

5 Introduction

In the previous (lecture 7) section we defined the local (nodal) and group properties:

1. Degree distribution
2. Centrality metrics

3. Similarity metrics and assortatitive networks

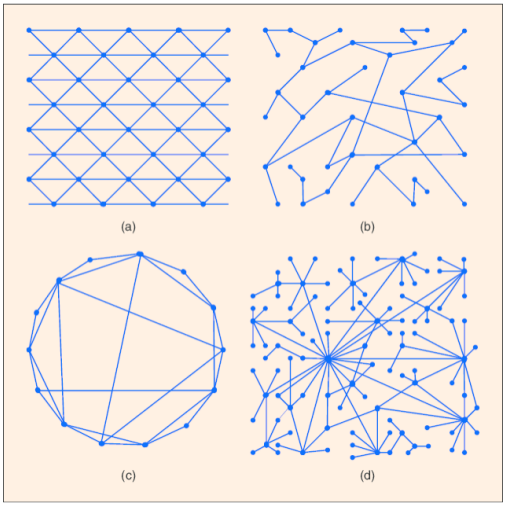
4. Clustering coefficient

5. Reciprocity

We also saw how helpful these graph features can be for understanding roles and special relationships in a network, and making predictions on networks.

In this section we will discuss random graphs.

6 Random Graphs



As is the case for random experiments we need to define a sample space Ω and probability measure $\mathcal{P}$ (on the probability space $(\Omega, \mathcal{F}, \mathcal{P})$). For random graphs this sample space is a set of graphs, and for simplicity we will consider the unweighted case, which makes this set discrete.

6.1 Why Do We Care?

Although these graphs are abstractions and not direct representations of reality, they can still have quite a bit of utility. Much of that utility comes from ways in which they can help us understand "real world" graphs. A few of the most important ways they can do this are:

- They show the idiosyncrasies of real graphs (somewhat) to the extreme
- They are a useful benchmarks for the properties of true networks
- They are a reasonably good fit and can help when making predictions

6.2 Random Graph Probability

Typically, we either fix the number of edges $M = |\mathcal{E}|$, the number of nodes $N = |\mathcal{V}|$, or both. When both N and M are fixed the graph density is fixed for all possible graph outcomes as:

$$\delta = \frac{|\mathcal{E}|}{\binom{N}{2}} = \frac{2M}{N(N-1)} \tag{28}$$

6.2.1 The Uniform Model

The name **Random Graph** is often associated with a **specific model**: a graph obtained choosing uniformly M pairs at random from $\mathcal{V}^2$ that form $\mathcal{E}$ such that $|\mathcal{E}| = M$. This setup specifies completely the random model, which we refer to using the notation $\mathscr{G}(N, M)$. We can also say that:

1. The sample space contains all possible graphs
2. The probability of each graph is uniform, so all graphs have probability equal to $1/|\mathscr{G}(N, M)|$

But what exactly is $|\mathscr{G}(N, M)|$?

We can figure that out by first noticing that there are $\binom{N}{2}$ possible edges and we need to chose M arbitrary combinations of them:

$$|\mathscr{G}(N, M)| = \#\text{ possible way of choosing } M \text{ elements our of } \binom{N}{2} \tag{29}$$

which implies:

$$|\mathscr{G}(N, M)| = \binom{\binom{N}{2}}{M} \quad \Rightarrow \quad P(\mathcal{G}) = \frac{1}{\binom{\binom{N}{2}}{M}} \tag{30}$$

We can interpret this as generating the $N \times M$ incidence matrix B uniformly at random.

6.2.2 Averages

If we consider a graph metric $\phi(\mathcal{G})$, we can find the average across all these graph samples using the graph distribution $P(\mathcal{G})$ to get

$$\mathbb{E}[\phi(\mathcal{G})] = \sum_{\mathcal{G} \in \mathscr{G}(N,M)} \ell(\mathcal{G}) P(\mathcal{G}) = \frac{1}{\binom{\binom{N}{2}}{M}} \sum_{\mathcal{G} \in \mathscr{G}(N,M)} \phi(\mathcal{G}) \tag{31}$$

When $N \gg 1$ and $M \gg 1$ these quantities tend to **harden**. Where by harden we mean that they approach a limit that depends on graph density (because of the law of large numbers).

6.3 Poisson/Erdös and Rényi Graphs

Fixing the number of edges makes computing the expectations and limits of these random graph features difficult. A more tractable model is one where the probability p that two nodes are connected is fixed, but the number of edges in $\mathcal{E}$ is not. We call such a model a **Poisson Random Graph** or a **Erdös and Rényi graph** and we will refer to it here as $\mathscr{G}(N, p)$.

We can interpret each graph formation as a Binomial experiment that picks a pair of nodes (from the $\binom{N}{2}$ possible pairs) possible with probability p. Let us indicate by $X \sim \mathcal{B}(K, p)$ a random variable whose distribution is Binomial with parameter p (probability of success) and K (number of trials). Thus, the probability of any given random graph with M edges is the same and is:

$$P(\mathcal{G}||\mathcal{E}| = M) = p^M (1-p)^{\binom{N}{2} - M} \tag{32}$$

where there are M successes (links) over $\binom{N}{2}$ possibilities. This means that $|\mathcal{E}| \sim \mathcal{B}(\binom{N}{2}, p)$ and all graphs with the same number of edges have equal probability $p^M (1-p)^{\binom{N}{2} - M}$.

This model greatly simplifies the calculation of several average graph features. One example of this is the **average number of edges**, which is just the mean of the Binomial random variable that defines the graph probability. This implies that the expected number of edges of an Erdös and Rényi graph $\mathscr{G}(N, p)$ is:

$$\mathbb{E}[|\mathcal{E}|] = \binom{N}{2} p \tag{33}$$

Since the expected value for the Binomial distribution $\mathcal{B}(K, p)$ is $\mathbb{E}[X] = Kp$.

We can then use the average number of edges to easily obtain the average degree. We already know that the average degree is:

$$\frac{1}{N} \sum_{v \in \mathcal{V}} d_v = 2 \frac{|\mathcal{E}|}{N} \tag{34}$$

from the **Handshaking** theorem. Thus, recalling that $\binom{N}{2} = \frac{1}{2} N(N-1)$, averaging over the distribution of $\mathscr{G}(N, p)$ gives:

$$\mathbb{E}\left[\frac{1}{N} \sum_{v \in \mathcal{V}} d_v \right] = 2 \frac{\mathbb{E}[|\mathcal{E}|]}{N} = \frac{2}{N} \binom{N}{2} p = (N-1)p \tag{35}$$

This is interesting, because it makes it clear that for the limit of the expected degree as $N \to +\infty$ to converge to a finite value > 0, p cannot vanish faster than $O(1/N)$.

We can further specify the degree distribution by defining its distribution directly as

$$p_k = \binom{N-1}{k} p^k (1-p)^{N-1-k} \tag{36}$$

which is a Binomial distribution with parameters $N-1$ and p. Put succinctly the distribution is $\mathcal{B}(N-1, p)$. The reasoning behind this is that the edges connecting each node to the remaining $N-1$ nodes is like the Bernoulli trial. Since we have $N-1$ of these trials the process becomes a Binomial experiment with probability of success (i.e. of an edge being added) equal to p.

7 Asymptotic Analysis

The Binomial distribution has two possible limits, for $N \to +\infty$:

1. the Gaussian distribution if p is constant, i.e. $d_v \sim \mathcal{N}(\mu, \sigma^2)$ with $\mu = pN$ and $\sigma^2 = p(1-p)N$
2. the Poisson distribution when $p = \frac{c}{N-1} \to 0$

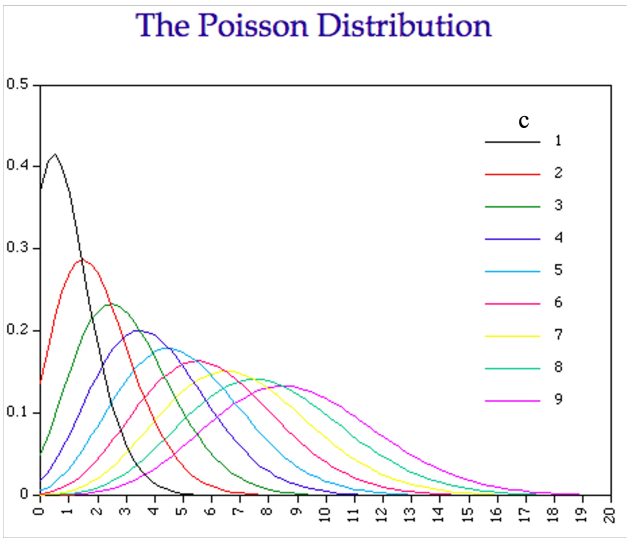
The trend for large $\mathscr{G}(N, p)$ networks has been widely studied, particularly for the case where the limit of the degree is a constant. This this case we say

$$p = \frac{c}{N-1}, \quad 0 < c < +\infty \quad \text{a constant} \tag{37}$$

Note: that $\to \mathbb{E}[d_v] = c$.

It is natural to be more interested in the second (Poisson) case, because in the first case where p is finite the network tends towards becoming a complete graph (in its asymptotic behavior). So let's take a look at the Poisson case! But first we should revise the definition of the distribution itself which is

$$N \gg 1 \quad p_k \to e^{-c} \frac{c^k}{k!} \tag{38}$$



We call such graphs $\mathscr{G}(N,p)$ **Poisson random graphs**. These graphs follow the behavior of the Poisson distribution, which means that it has exponentially decaying tail and is unimodal with mode c, as can be seen in 7.

7.1 Clustering Coefficient of $\mathscr{G}(N,p)$

Recall that the clustering coefficient measures the transitivity of nodes with respect of being connected. Since the probability of being connected to any two nodes is the same and is p then

$$\mathbb{E}[C] = p \tag{39}$$

In the asymptotic regime, this means that $C = \frac{c}{N-1} \to 0$.

This is one of the most striking differences between $\mathscr{G}(N,p)$ and networks like social networks, where friends of friends tend to be friends.

7.2 Connectivity and Giant Components

A giant component is a graph component that includes most of the nodes N. Let us call such a component $\mathcal{V}^g \subseteq \mathcal{V}$. Now we can ask whether there is a value of p that leads (with high probability) to the emergence a giant component in all possible outcomes of $\mathscr{G}(N,p)$?

We can calculate this value by studying the expected fraction of nodes γ that are in the giant component, or more formally

$$\gamma := \frac{\mathbb{E}[|\mathcal{V}^g|]}{N} \equiv P(\text{a randomly chosen node } v \in \mathcal{V}^g) \tag{40}$$

so $1-\gamma = P(\text{a randomly chosen node } u \notin \mathcal{V}^g)$.

7.2.1 Probability v is not in the Giant Component

The probability that a node v is not in $\mathcal{V}^g$ is equal to the probability of the event that v has no edge connecting it to a nodes in $\mathcal{V}^g$. This can happen if, considering a randomly chosen node $u \neq v$:

1. **Event 1**: u is not connected to node v. This event has probability $1-p$
2. **Event 2**: $u \notin \mathcal{V}^g$ and it is connected to node v. This event has probability $(1-\gamma)p$

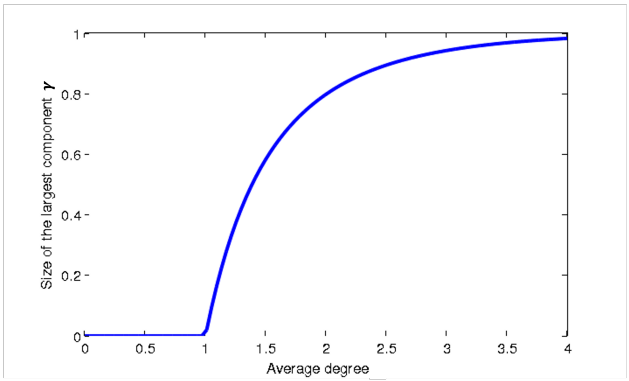


Figure 3: Phase Transition Graph

These events are mutually exclusive, so we can add their respective probabilities to obtain:

$$P(u \neq v \text{ Event 1 happens or Event 2 happens}) = 1 - p + (1-\gamma)p \tag{41}$$

From this we can derive the following equation:

$$\begin{aligned}
1 - \gamma &= P(\text{a randomly chosen node } v \notin \mathcal{V}^g) && (42)\\
&= P(\forall u \neq v \text{ Event 1 happens or Event 2 happens}) && (43)\\
&= \prod_{u \in \mathcal{V}, u \neq v} P(u \neq v \text{ Event 1 happens or Event 2 happens}) && (44)\\
&= (1 - p + (1-\gamma)p)^{N-1} && (45)\\
&= (1 + (1 - \gamma - 1)p)^{N-1} = (1 - \gamma p)^{N-1} && (46)
\end{aligned}$$

where the third equality exists because the events for each node $u \neq v$ are independent and the next equation uses the expression we found in the previous slide, simplified in the last equation.

With some further algebra, substituting $p = \frac{c}{N-1}$ we have:

$$N \gg 1 \quad \Rightarrow \quad 1 - \gamma \approx e^{-c\gamma} \quad \Rightarrow \quad \gamma \approx 1 - e^{-c\gamma} \tag{47}$$

We cannot solve for γ explicitly, but it is relatively simple to get a numerical solution to find γ such that $\gamma \approx 1 - e^{-c\gamma}$.

Note that the solution is $\gamma = 0$ iff

$$1 = d(1 - e^{-c\gamma})/d\gamma|_{\gamma=0} = ce^{-c\gamma}|_{\gamma=0} = c \tag{48}$$

This implies that there is a **phase transition** such that for $c \leq 1$ there is no giant component, while there is always a giant component (a component with $O(N)$ nodes for $c > 1$).

Figure 7.2.1 shows how the percentage size of the giant component γ grows with the average degree c.

7.3 Average Path Depth

To compute the average path length we can use the following argument:

- Imagine we look at the formation of the network step by step, starting by connecting one node at random, then a neighbor, then one its neighbors etc.
- Every time we add new edges to the network, we add on average $\mathbb{E}[d_v] = (N-1)p$ edges. For $p = \frac{c}{N-1}$ we are adding $\mathbb{E}[d_v] = c$ edges every time.

This implies that the average number of nodes that are s steps away must be c^s. This must stop when $c^s = N$, because there are N nodes in the network.

By taking the natural logarithm at both sides of the equality $c^s = N$ we get:

$$s \approx \frac{\ln N}{\ln c} \tag{49}$$

Let d_{uv} be the shortest path between nodes u and v. A more rigorous calculation for $N \gg 1$ (we will skip the details for brevity) yields the following result:

$$P(d_{uv} > \ell) = \left(1 - \frac{c}{N}\right)^{c^{\ell-1}} \to e^{-\frac{c^\ell}{N}} \tag{50}$$

For $P(d_{uv} > \ell) \to 1$ $c^\ell = aN^{1+\epsilon}$ where a is some scaling factor and $\epsilon > 0$ is a small positive constant. Thus, taking the logarithm at both sides of $c^\ell = aN^{1+\epsilon}$, we have:

$$\ell = \overbrace{\frac{\ln a}{\ln c}}^{A} + (1+\epsilon)\frac{\ln n}{\ln c} \tag{51}$$

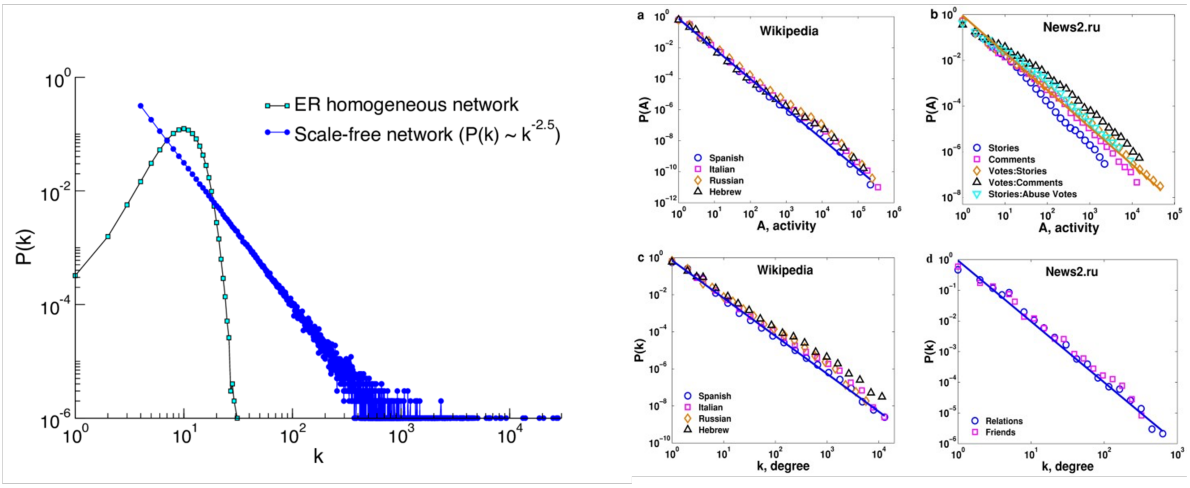
which, up to a constant, is similar to what we found before.

7.4 How Realistic is $\mathcal{G}(N,p)$?

We have seen that $\mathcal{G}(N,p)$ allows for interesting and tractable analysis of the graph features, but "how realistic is $\mathcal{G}(N,p)$?"

The answer is: not that much, in most cases. Networks have more structure, even though they may be very different from one another. This is readily apparent from comparing features like the degree distribution. The only feature whose trend is realistic is actually the average path length, which tends to be on the order of $\ln N$.

7.4.1 Comparison of $\mathcal{G}(N,p)$ Features and Real Networks



On the right we see a comparison of degree distributions of the bla network and that of $\mathcal{G}(N,p)$ (which is Binomial) with a graph that has a power law degree distribution. On the left we see the degree distribution in semi-logarithmic scale for the Wikipedia pages of different countries.

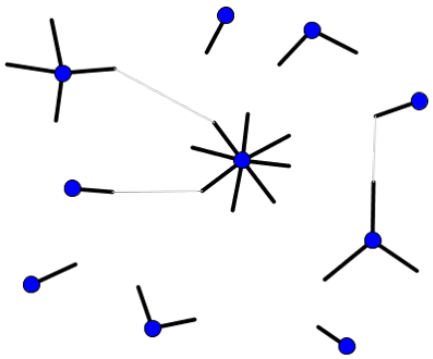
8 The Configuration Model

The configuration model is another random graph model that is similarly tractable but more realistic. The basic idea is to fit an arbitrary degree sequence or degree distribution to the sample space of graphs. From this basic setup we get two versions:

1. In one the degree sequence is fixed d
2. In the second one, the degree distribution p_k is given, and the degrees are drawn at random

This model allows, for example, to match the power law trends we visualized in the previous slides

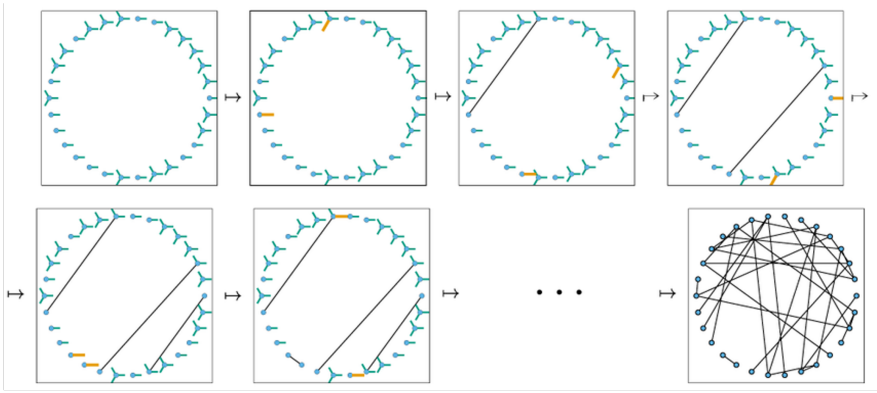
8.1 Generation of the Graph for the Configuration



The generation algorithm goes as follows:

1. The N nodes are assigned the degrees, forming a series of stubs
2. The **stubs are matched** by connecting one of the edges each time
3. The matching stubs are chosen uniformly at random from the set of those that have yet to be matched

8.2 Forming the Random Graph



This is an iterative process that can work **only** if the sum of the degree sequence is even, because of the handshaking theorem. That means $d^\top 1 = 2|\mathcal{E}|$ must be satisfied. If the degree sequence is drawn at random, we can throw away those that do not meet this criterion.

8.3 Edge Probability

Recall $M = |\mathcal{E}|$, so there are $2M$ stubs total to match. The probability of a match between node u stub with a specific edge node v stub chosen at random uniformly is $d_v/2M - 1$. Adding the probability of the events that correspond to all possible d_u choices for the edge, the probability of an edge between u and v is:

$$p_{uv} = \frac{d_u d_v}{2M - 1} \tag{52}$$

8.4 Multiedges and Self-Edges

Nothing prevents us from choosing node v again to match to another edge of node u. That means we can create a multiedges. Since one is already occupied, the probability of a second edge between (u, v) is:

$$p_{uv}^{(2)} = \frac{d_u d_v}{2M - 1} \times \frac{(d_u - 1)(d_v - 1)}{2M - 1} = O(M^{-2}) \tag{53}$$

which is relatively small for large M relative to the first edge which is $O(M^{-1})$.

It can also be shown that the probability of a self loop is:

$$p_{uu} = \frac{d_u(d_u - 1)}{2(2M - 1)} \tag{54}$$

so this is not that much smaller if a node has a high degree.

8.5 The Friendship Paradox

One peculiar property of the configuration model is the *friendship paradox*: the degree of a node tends to be less than the average degree of its neighbors. To get a feel for this consider that the probability of choosing at random a node u with degree $d_u = k$ is p_k of the degree distribution. The probability that the node u is also attached to a node with degree k is $k/(2M - 1)$ where $2M = \sum_{v \in \mathcal{V}} d_v$.

Consider the event: $Event_k$=a node with $d_u = k$ attaches to one with $d_v = k$. The expected numbers of nodes with degree k is Np_k and so:

$$P(Event_k) = \frac{k}{2M - 1} N p_k \approx \frac{k p_k}{d^{\text{ave}}} \tag{55}$$

where $d^{\text{ave}} = 2M/N$ is the average degree (we ignore -1, assuming $M \gg 1$). It follows that

$$\mathbb{E}[d_v | d_u = k, (u, v) \in \mathcal{E}] = \sum_k k P(Event_k) = \frac{1}{d^{\text{ave}}} \sum_{k=1}^{N} k^2 p_k \tag{56}$$

$$= \frac{1}{d^{\text{ave}}} (\sigma_d^2 + (d^{\text{ave}})^2) = \frac{\overbrace{\sigma_d^2}^{>0}}{d^{\text{ave}}} + d^{\text{ave}} > d^{\text{ave}} \tag{57}$$

Even though the configuration model does not achieve the exact desired number, this phenomenon is observed in real networks as can be seen in 8.5.

Network	n	Average degree	Average neighbor degree	$\frac{\langle k^2 \rangle}{\langle k \rangle}$
Biologists	1 520 252	15.5	68.4	130.2
Mathematicians	253 339	3.9	9.5	13.2
Internet	22 963	4.2	224.3	261.5

Figure 4: Empirical Friendship Paradox Survey

8.6 Clustering Coefficient

Similar to the friendship paradox calculation, we can calculate the probability that a node with degree k has degree $k+1$, which is called the **excess distribution**:

$$q_k = P(Event_{k+1}) \approx \frac{(k+1)p_{k+1}}{d^{\text{ave}}} \tag{58}$$

A node v is in a triangle with nodes u and i that have degree d_u and d_i if:

- u and i have one of their other edges in common, which occurs with probability $\frac{(d_u-1)(d_i-1)}{2M}$
- and v has them as neighbors, which occurs with the product of the respective excess probability $q_{d_u-1}q_{d_v-1}$

With some algebra, and by performing the change of variables $d_u - 1 = k_u$ and $d_i - 1 = d_v$ the average of this probability is

$$\mathbb{E}[C] = \sum_{k_u,k_i=0}^{d^{\max}-1} q_{k_u} q_{k_u} \frac{k_u k_i}{2M} = \ldots = \frac{1}{N} \frac{\mathbb{E}[k^2] - \mathbb{E}[k]}{\mathbb{E}[k^3]} \tag{59}$$

For exponentially decaying distributions this is going to be small ($O(1/N)$), but that is not so for power law degree distributions $p_k \propto k^{-\alpha}$.

8.7 Random Graphs with Given Expected Degree

Can we modify $\mathscr{G}(N,p)$ to impose a degree distribution other than the Binomial? The answer is yes, but only if we appropriately select the edge probabilities p_{uv} instead of assuming that it is a constant p. This is called the Chung-Lu model and the choice of p_{uv} that is used is as follows:

$$p_{uv} = \begin{cases} \frac{c_u c_v}{2M} & u \neq v \\ \frac{c_u^2}{4M} & \text{else} \end{cases} \tag{60}$$

In this case the average number of edges can be shown to be M and the expected degree is $\mathbb{E}[d_u] = c_u$. Let's quickly end by proving that M is the average number of edges. We can do this by working through the sum of connecting probabilities:

$$\sum_{u<v} p_{uv} + \sum_u p_{uu} = \sum_{u<v} \frac{c_u c_v}{2M} + \sum_u \frac{c_u^2}{4M} = \sum_{uv} \frac{c_u c_v}{4M} = M \tag{61}$$

9 Conclusion

- We surveyed a couple important stochastic models for graphs

- The Poisson random graph is a good benchmark
- The configuration model allows for more realism
- Next time we will consider a few more models and compare their average features with real networks