Network Types

Written by James D. Wilson

University of San Francisco

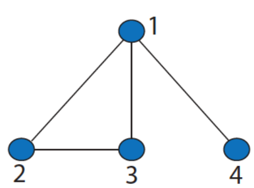
Static Networks

A **static network** is a network used to represent a single snapshot of a relational system. A static network model can be represented by the mathematical object

$$G = (V,\ W),$$

where V is the **vertex set** describing the actors of the system, and $W = \{W_{uv} : u, v \in V\}$ is the collection of **edge weights** describing the strength of relationships between each pair of vertices. A network is considered **unweighted** if for all $u, v \in V$, $W_{uv} \in \{0, 1\}$, where $W_{uv} = 1$ if there is an edge between nodes u and v and 0 otherwise.

Let $|V|$ denote the cardinality of the vertex set V, namely, the number of vertices in G. Then G can be equivalently represented by the $|V| \times |V|$ **adjacency matrix** $\mathbf{A} = [A_{uv}]$, where $A_{uv} = W_{uv}$ is the edge weight between nodes u and v. It is readily observed that $\mathbf{A}$ requires $|V|^2$ entries to be stored in memory, which can be problematic for large networks (when $|V|$ is of the order of millions). In cases like these, we can encode G using a sparse representation via the edge list $\mathbf{E}$. Rows of $\mathbf{E}$ contain the entries u, v and W_{uv} only for those pairs of nodes such that $W_{uv} > 0$. See Figure 1 for a toy example.



$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

$E =$

1	2	1
1	3	1
1	4	1
2	3	1

Figure 1: A toy example demonstrating three representations of a network: the raw network, an adjacency matrix A, and an edge list E.

Practical Note 1. For computational purposes, we must either use the adjacency matrix $\mathbf{A}$, or its sparse edge list encoding $\mathbf{E}$. In the case of large networks, one should utilize the edge list whenever possible.

Undirected Networks

In an **undirected network**, relationships between actors in the system are mutual. That is, edges between two nodes represent a symmetric relationship. For example, in a Facebook friendship network undirected, edges between two people imply that they are Facebook friends with one another.

A graph G that is undirected has the special property that its associated adjacency matrix is *symmetric*. That is, $A_{uv} = A_{vu}$. It follows that we only need to consider $\binom{|V|}{2}$ edges.

A special case of an undirected network is a **tree**. A tree is simply a graph where any two vertices contain exactly one path between them. Another way to characterize a tree is as an undirected network that contains no cycles, or loops leading from one vertex back to itself.

Directed Networks

In a **directed network**, edges between nodes have a direction associated with them. As a consequence, relationships in the system need not be mutual. In the school children network from Moreno's sociograms, for example, a directed edge from node A to node B suggests that child A views child B as a friend. Directed networks are common in biological networks, where, for example, a directed edge may represent a gene's regularatory control over another gene.

The adjacency matrix of a directed graph G is not *symmetric*. As a result, one must consider all $|V|^2$ possible edges in the network for analysis.

A special type of directed network is an **directed acyclic graph (DAG)**. A **cycle** in a directed graph is a closed loop of edges with arrows on each of the edges pointing in the same direction. A DAG is a directed graph that does not contain any cycles. These networks are particularly useful for describing the gene-gene interactions in a biological network.

Bipartite Networks

A **bipartite network** (or **bigraph**) is a special type of static network whose vertex set V can be divided into two disjoint sets V_1 and V_2 in such a way that every edge connects a vertex in V_1 to one in V_2.

The two sets V_1 and V_2 may be thought of as a coloring of the network with two colors: if one colors all nodes in V_1 blue, and all nodes in V_2 red, each edge has endpoints of differing colors. See Figure 2 for an illustration.

Bipartite networks are commonly used for *recommendation systems*. Consider the movie recommendation problem where one wants to suggest movies to viewers according to their interest in other movies. A bipartite network naturally represents this system, where V_1 represents viewers and V_2 represents the movies that may or may not be suggested. We will revisit this problem later in these notes.

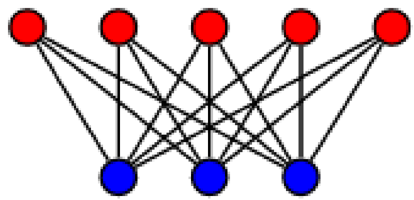


Figure 2: An example of a full bipartite graph, where the vertices of each disjoint set are colored either red or blue. Edges must be so that they never connect two nodes of the same color. Image reproduced from this website.

Multilayer Networks

A **multilayer network** is a collection $\mathbf{G}_m = (G_1, \ldots, G_m)$ of m networks. The graph G_ℓ is commonly referred to as the ℓth layer of the network. The layers $G_\ell = (V_\ell, W_\ell)$ have vertex sets V_ℓ and edge weights W_ℓ that may vary from layer to layer. Layers are regarded as unordered so that the indices $\ell \in [m]$ do not reflect an underlying spatial or temporal order among the layers.

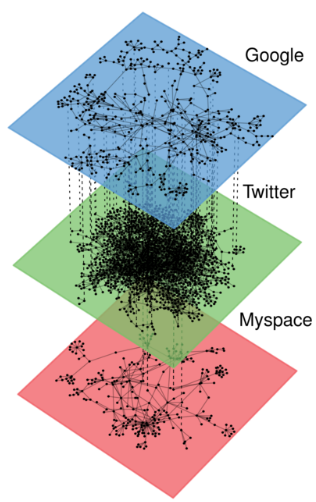


Figure 3: An example of a multilayer social network where layers represent different modes of communication. Image reproduced from this website.

Multilayer network models have been applied to a variety of problems, ranging from the modeling and analysis of ground and air transportation [1, 4] to the study of effects of social interactions on economic exchange [2]. As mentioned in Section 2, multilayer networks are also commonly used in the field of network neuroscience. Figure 3 shows an example of a multilayer social network.

When computationally analyzing multilayer networks, one must deal with a collection or list of adjacency matrices $\{\mathbf{A}_1, \ldots, \mathbf{A}_m\}$ or edge lists $\{\mathbf{E}_1, \ldots, \mathbf{E}_m\}$, which can very quickly lead to memory issues in a stand-alone computing environment. However, perhaps the most efficient way to handle multilayer networks is through what I'll call the "super-edgelist" $\mathcal{E}$, which is a list containing the following columns describing $\mathbf{G}_m$: $\{node1, layer1, node2, layer2, weight\}$. In this way, we can efficiently store inter- and intra- layer connections, as illustrated in Figure 3.

Practical Note 2. When the layers of $\mathbf{G}_m$ contain no inter-layer connections - namely when nodes of one layer is never connected nodes of a different layer - then $\mathbf{G}_m$ is generally referred to as a **multiplex network**.

Practical Note 3. When $V_\ell \equiv V$ for all layers ℓ, then $\mathbf{G}_m$ can alternatively be represented through a static graph with multiple edge types known as a **hypergraph**. Despite this simplification, it is still necessary to keep up with the types of edges between each pair of nodes and so storage of such a network still requires a "super-edgelist" representation as described before.

Dynamic Networks

Dynamic networks is a sequence graphs $\mathbf{G}_T = (G_1, \ldots, G_T)$, where there is a temporal ordering of the networks. Dynamic networks are a special case of multilayer networks, where there is a dependence between sequential networks G_{t-1}, G_t of the sequence. Figure 4 illustrates an example of a dynamic network representing changes in the structure of co-voting habits among member of the U.S. Senate over time.

Dependencies in a dynamic network sequence may be represented by, for example, a Markov chain representation or some other autoregressive or moving average dependency. The study of dynamic networks is still very much in its early stages. [3] provides a recent textbook-level treatment of dynamic network analysis to date.

Like multilayer networks, one must also store a list of T adjacency matrices or edge lists for computation on dynamic networks. Moreover, one must also store information describing the dependencies between networks in the sequence and so the super-edgelist representation is often the preferred method of storage.

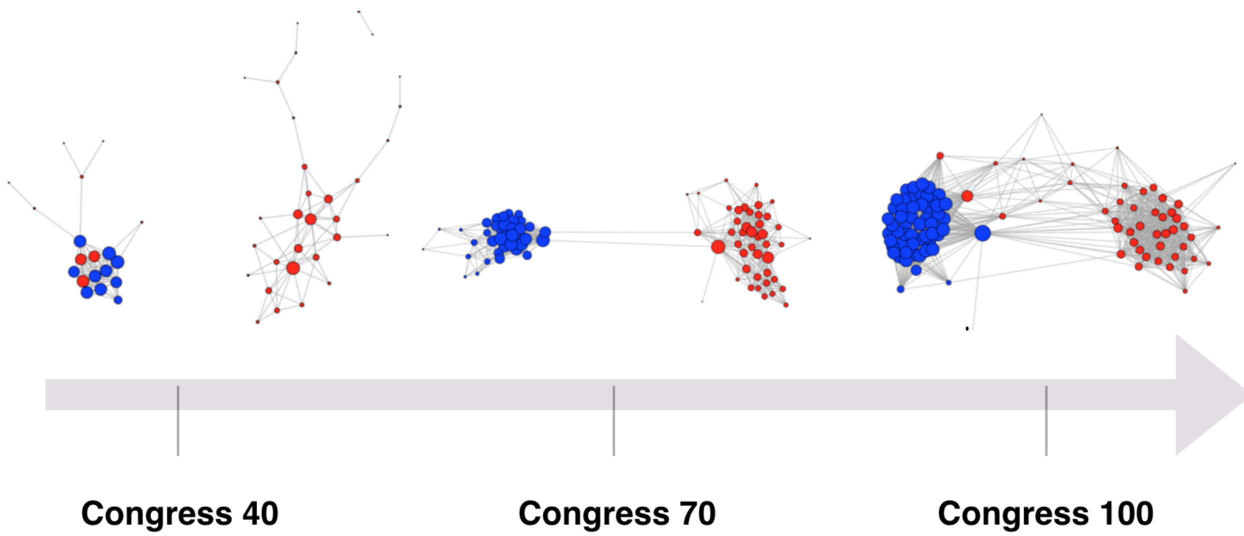


Figure 4: A dynamic network representation of the voting behavior among Senators in the U.S. Congress. Original image from [5].

References

[1] Alessio Cardillo, Jesús Gómez-Gardeñes, Massimiliano Zanin, Miguel Romance, David Papo, Francisco Del Pozo, and Stefano Boccaletti. Emergence of network features from multiplexity. *Scientific reports*, 3, 2013.

[2] Simone Ferriani, Fabio Fonti, and Raffaele Corrado. The social and economic bases of network multiplexity: Exploring the emergence of multiplex ties. *Strategic Organization*, 11(1):7–34, 2013.

[3] Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.

[4] Emanuele Strano, Saray Shai, Simon Dobson, and Marc Barthelemy. Multiplex networks in metropolitan areas: generic features and local effects. *Journal of The Royal Society Interface*, 12(111):20150651, 2015.

[5] James D Wilson, Nathaniel T Stevens, and William H Woodall. Modeling and estimating change in temporal networks via a dynamic degree corrected stochastic block model. *arXiv preprint arXiv:1605.04049*, 2016.