LINKS Workshop 2023 - Intermediate SNA

Eric Hsienchen Chu

June 8, 2023

This Reading Note is a part of LINKS Workshop 2023 - Intermediate SNA, June 5 - June 9, 2023. The materials and contexts are greatly from Borgatti et al. (2022) and are preserved as resources for future SNA work. I thank CALL-ECL Project for their generous guidance and funding support. All reports of typos/errors are welcome.

Contents

1 Introduction - Why Networks? **2**

2 Mathematical Foundations **2**
- 2.1 Graphs . . . 2

3 Research Design **4**

4 Data Collection (skip) **5**

5 Data Management in R (skip) **5**

6 Multivariate Techniques in Network Analysis **5**
- 6.1 Multidimensional Scaling (MDS) . . . 5
- 6.2 Correspondence Analysis (CA) . . . 5
- 6.3 Hierarchical Clustering . . . 5

7 Visualization in R **6**

8 Local Node-level Measures **8**
- 8.1 Tie composition . . . 8

9 Centrality **9**
- 9.1 Degree Centrality . . . 9
- 9.2 Closeness Centrality . . . 10
- 9.3 Betweenness Centrality . . . 12

10 Group-level Measures **13**

1 Introduction - Why Networks?

Overview. General framework:

- Dyad level ($O(n^2)$): study pairwise relations between actors, e.g., whether have ties (basic).
- Node level ($O(n^1)$): study aggregations of dyad-level measurements, e.g., counting the number of ties
- Network level ($O(n^0)$): e.g., well-connected networks tend to diffuse ideas faster?

Definition 1.1 (Interactions and Flows). Interactions are behaviors with respect to others and (often) observable by third parties. Flows are the outcomes of interactions.

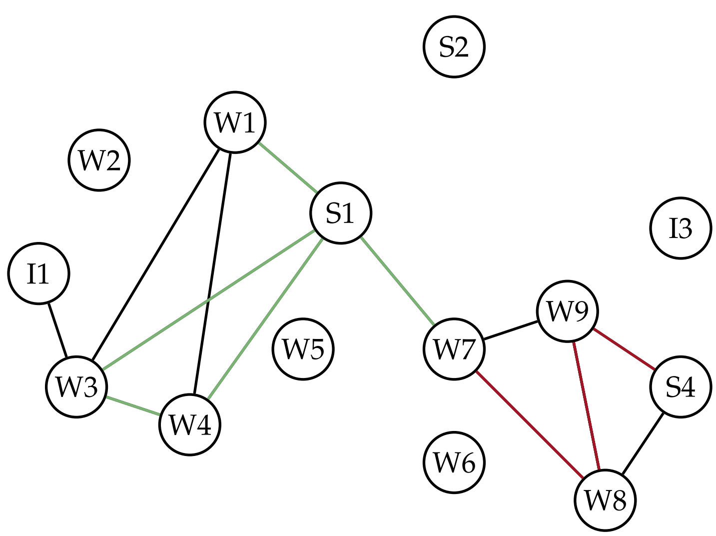
Remark. Goal of SNA is to identify and describe the structure of network or capture aspects of individual's positions in the network.

2 Mathematical Foundations

2.1 Graphs

Motivation. All the instances of network analysis have in common is the branches of graph theory and matrices.

Definition 2.1 (Graph). A graph $G(V, E)$ consists of a set of vertices V (i.e., nodes) and a set of edges E (i.e., links). We denote $(i, j) \in E(G)$ as the vertex i and j are connected by an edge in graph G. Graph can be directed or undirected.

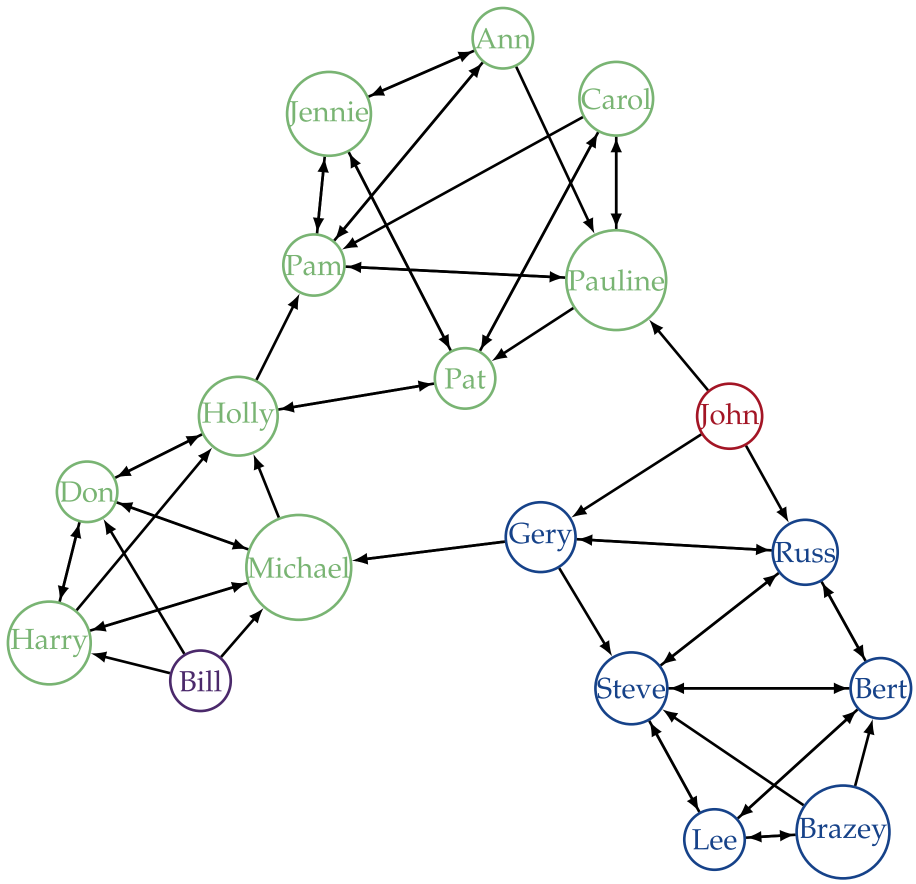


Replication. Figure 2.2 in Borgatti et al. (2022).

Remark. Path never revisits a node, e.g., $S4 - W9 - W8 - W7$ is a sequence of path. Trail never revisits an edge, e.g., $W1 - S1 - W3 - W4 - S1 - W7$ is a sequence of trail.

Definition 2.2 (Degree)**.** The number of connections a node has is denoted as *degree*. Nodes with degree 0 are "isolates," and nodes with degree 1 are "pendants."

Definition 2.3 (Components)**.** A *component* is defined as a maximal set of nodes in which every node can reach every other by some path. The "maximal" term means that we must include the specific node if adding it to the set would not violate the condition that every node can reach everyone.



Replication. Figure 2.3 in Borgatti et al. (2022).

Remark. Color signals components. We notice the set {Gery, Russ, Steve, Burt, Lee, Brazey} represents one component. Why? We cannot not include more nodes to *this* set while satisfying the condition that each node connects to everyone. It is thus a 'maximal' set.

Definition 2.4 (Adjacency matrix)**.** The *adjacency matrix* **X** of a non-valued graph is defined as a matrix in which the entry $x_{ij} = 1$ if existing a tie from i to j and $x_{ij} = 0$ if not (direction matters).

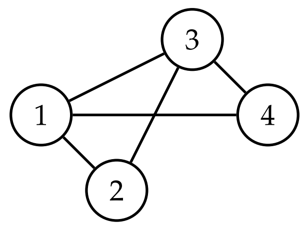
Definition 2.5 (Compound relations)**.** The **inner product** of two adjacency matrices constructs compound social relations.

Example 2.1. Suppose matrix **F** represents "friend of" relations and matrix **E** represents "enemy of" relations, then the inner product **FE** is a compound relation meaning "enemy of a friend of," where $\mathbf{FE}(i, j)$ indicates the number of i's friends who have j as an enemy.

Definition 2.6 (Application of Matrix Transpose: Undirected Graph)**.** Suppose we have a network matrix $\mathbf{A}$, then the diagonal of inner product $\mathbf{A} \cdot \mathbf{A}^T$ yields the count of ties for each node, where entry (i, j) yields the count of node both node i and j connect to.

Remark. For Directed Graph: $\mathbf{A} \cdot \mathbf{A}^T$ yields out-degree ties, and $\mathbf{A}^T \cdot \mathbf{A}$ yields in-degree.

Example 2.2. Consider the following network:



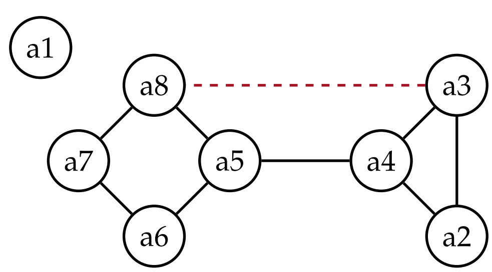
We specify the adjacency matrix $\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$, then we get $\mathbf{A} \cdot \mathbf{A}^T = \begin{bmatrix} 3 & 1 & 2 & 1 \\ 1 & 2 & 1 & 2 \\ 2 & 1 & 3 & 1 \\ 1 & 2 & 1 & 2 \end{bmatrix}$, where $\{3, 2, 3, 2\}$ counts ties and 2 signals that Node 1 & 3 have two common connections (i.e, to Node 2 & 4).

3 Research Design

Overview. Researchers care about the reliability of sources of data and their validity issues. Essentially, four kinds of errors are tested: ① omission of nodes, ② omission of ties, ③ inclusion of false nodes, and ④ inclusion of false ties. Researchers have found that errors in various centrality measures resulting from random exclusion and inclusion of edges in random graph vary as a function of characteristics of the network.

Summary. Several key errors in SNA research:

- Omission & commission errors. Missing/Adding nodes or edges can obviously over/underestimates the centrality measures. Particularly common for open-ended survey. For instance, falsely adding a tie between $a3$ and $a8$ (the dashed line) can cause the two nodes to be exaggeratedly important than they were.



Replication. Figure 3.2 in Borgatti et al. (2022).

- Attribution errors for edge/node. False attributions can yield non-exist networks (e.g., treating co-attendance as a tie).

Check Borgatti et al. (2022) pp.43-45 for more data/model specification errors.

4 Data Collection (skip)

5 Data Management in R (skip)

6 Multivariate Techniques in Network Analysis

6.1 Multidimensional Scaling (MDS)

6.2 Correspondence Analysis (CA)

6.3 Hierarchical Clustering

7 Visualization in R

Overview. We require ① *sna* or ② *igraph* packages for network visualization.

```
library(sna)
library(igraph)

# Read in the data
# First create an empty list object to store ``Krackhardt_HighTech`` networks:
Krackhardt_HighTech = list()

# Now we read in the data and add different elements to it:
Krackhardt_HighTech$Advice = as.matrix(
  read.csv("Krackhardt_HighTech_Advice.csv",
           stringsAsFactors=FALSE, row.names=1)
  ) # Same process for ``Advice``, ``Friendship``, ``Attribute`` (already matrix)
```

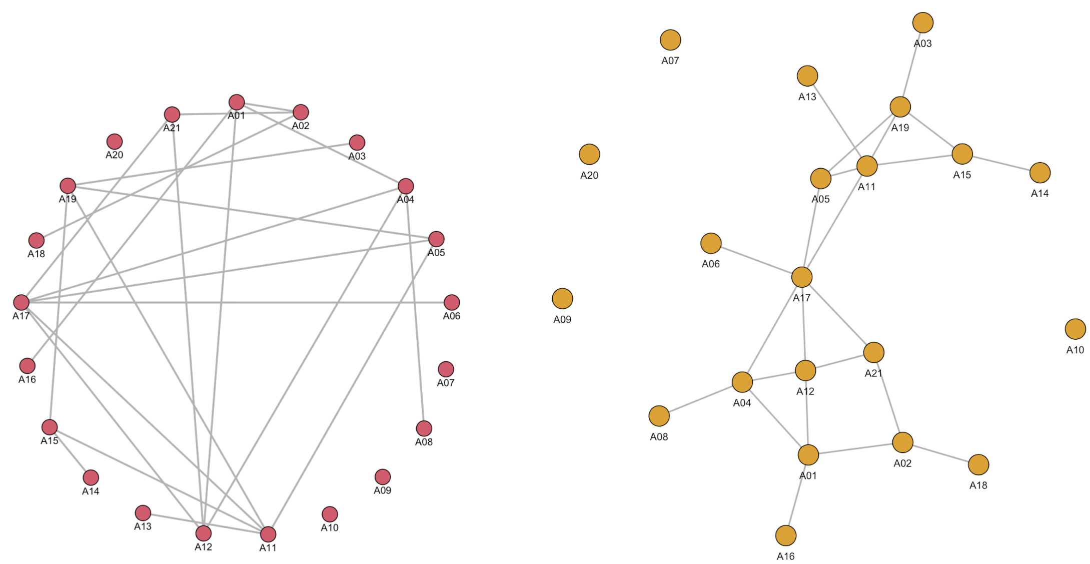
From **Definition 2.6**, we use the fact that $\mathbf{A} \cdot \mathbf{A}^T$ tells us info about the network (# of ties, common connections, etc). We can borrow the **t** function for such matrix transpose.

```
# Symmetrizing the network matrix: ``Krackhardt_HighTech``
# t: Matrix Transpose
Krackhardt_HighTech$Friendship_SymMin =
    (Krackhardt_HighTech$Friendship)*t(Krackhardt_HighTech$Friendship)
```

The following two chunks plot the network with two different packages.

```
# Figure 7.1: using the "sna" package
gplot(Krackhardt_HighTech$Friendship_SymMin,
      gmode = "graph",         # type of network: undirected #digraph: directed
      mode = "circle",         # how the nodes are positioned
      vertex.cex = 0.8,        # the size of the node shapes
      displaylabels = TRUE,    # to add the node labels
      label.pos = 1,           # to position the labels below the node shapes
      label.cex = 0.8,         # to decrease the size of the node labels
      edge.color = "grey70")
```

```r
# Figure 7.2: using the "igraph" package
KHF_SymMin_i = graph_from_adjacency_matrix(Krackhardt_HighTech$Friendship_SymMin,
                                           mode = "undirected", diag = FALSE)
plot(KHF_SymMin_i,
     #layout = layout_in_circle,
     vertex.size = 8,                # the size of the node shapes
     vertex.label.cex = 0.8,         # to decrease the size of the node labels
     #vertex.label.family = "",      # change the font (default  Times New Roman)
     vertex.label.color = "black",   # change the label colors (default blue)
     vertex.label.dist = 1.1,        # change the dist of the labels (default 0)
     vertex.label.degree = pi/2,     # to pos the labels below the node shapes
     edge.color = "grey70",
     edge.width = 2)                 # increase the width of the ties.
```



Note. The left plot (7.1) uses *sna*, the right one (7.2) uses *igraph*.

Remark. Always do the `graph_from_adjacency_matrix()` to transform data into “graph“ for *igraph* package. During the LINKS Workshop, one attendee mentioned that `library(threejs)` generates powerful visualization for networks (data in *igraph*). Here is an example with sample codes.

8 Local Node-level Measures

8.1 Tie composition

9 Centrality

Overview. Key highlights of centrality measures:

- Group-level: *"How centralized is the group?"* density, centralization, **homophily**, components
- Individual-level: *"How central is a node in the group?"* **degree centrality**, **closeness centrality**, **betweenness centrality**, structural holes, diversity of alters
- Dyad-level: *"What is the shortest distance between two nodes (A and B)?"* direct tie between two units, **level of structural equivalence**, geodesic distance, Number of shared clique membership

9.1 Degree Centrality

Motivation. Who has the most direct connections?

Definition 9.1 (Degree Centrality)**.** *Degree Centrality* is defined as the number of ties of a given type that a node has. For adjacency matrix $\mathbf{X}$, the degree centrality is the row/column sum, i.e., $d_i = \sum_j x_{ij}$.

Remark. Nodes with higher degree centrality tend to be more important in a network. Degree centrality can be an index of exposure of this node and is particularly important for some "directly influenced network.". In R, we can use `xDegreeCentrality()` for simple calculation.

Definition 9.2 (Eigencentrality)**.** *Eigencentrality* counts the number of nodes adjacent to a given node but weights each node by its centrality.

$$e_i = \frac{1}{\lambda} \sum_j x_{ij} e_j$$

where e is the eigencentrality score and λ is an eigenvalue (constant).

Remark. Eigencentrality says that each node's (eigen)centrality (e_i) is proportional to the sum of (eigen)centralities (e_j) of the nodes it is adjacent to (those $x_{ij} = 1$). By convention, we use the largest eigenvalue for eigencentrality and normalized it *s.t.* sum of squares $= 1$. We can use the function `xEigenvectorCentrality()`.

Motivation. What if there can be a repeated influence among nodes?

Definition 9.3 (Bonacich Beta Centrality)**.** *Bonacich Beta Centrality* is given by the equation

$$(\mathcal{I} - \beta \mathbf{X})^{-1} \mathbf{X} = \sum_{k=1}^{\infty} (\beta^{k-1} \mathbf{X}^k) = \mathbf{X} + \beta \mathbf{X}^2 + \beta^2 \mathbf{X}^3 + \cdots$$

Example 9.1 (Sample R Codes: Bonacich Beta Centrality)**.** Note that we normalize the Beta Centrality by ensuring the sum of centrality is 1.

```
# igraph
GRAPH11 = graph_from_adjacency_matrix(MAT11, mode = c("undirected"), diag = F)

# Sample output
bonpow(GRAPH11, exponent = 0.1)/sum(bonpow(GRAPH11, exponent = 0.1))}.
[1]  0.05570 0.055707 0.055707 0.182418 0.055707
[6] 0.055707 0.055707 0.182418 0.158428 0.095475 0.047013

bonpow(GRAPH11, exponent = 0.5)/sum(bonpow(GRAPH11, exponent = 0.5))
[1]  0.06779 0.067796 0.067796 0.152542 0.067796
[6] 0.067796 0.067796 0.152542 0.169491 0.084745 0.033898

max(eigen(MAT11)) # Generally, we can use this eigenvalue as beta
```

Remark. What β should we use here? When $\beta = 0$, this Bonacich Beta Centrality converges back to Degree Centrality. Generally, larger the better & we can use `max(eigen())` for such β (somewhat back to eigencentrality).

9.2 Closeness Centrality

Motivation. Who is most central- If you have information that you want *everyone* in the group to have, and you can only give it to one person, who would you give it to?

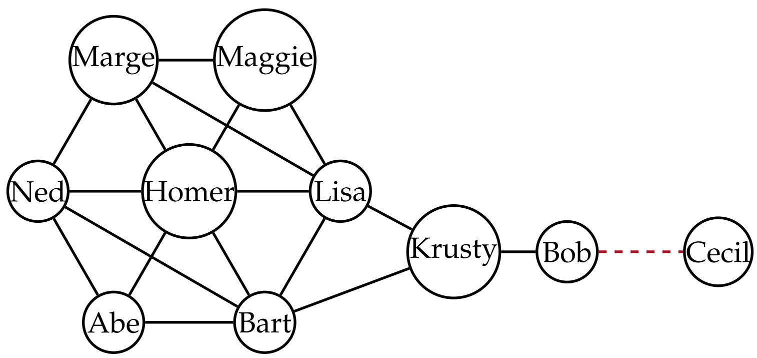
Definition 9.4 (Freeman's Closeness Centrality)**.** Closeness Centrality is normalized by $C_i = \frac{N-1}{\sum\limits_j geodesic(i,j)}$, where $N-1$ is the theoretical minimum farness of the network of N nodes.

Example 9.2 (Sample R Codes: Freeman's)**.** Below are some Freeman's Closeness Centralities using *sna* and *igraph*.

```
sna::closeness(dat, g = 1, nodes = NULL, gmode = "digraph",
                    diag = FALSE, tmaxdev = FALSE, cmode = "directed",
                    geodist.precomp = NULL, rescale = FALSE,
                    ignore.eval = TRUE) # default
# Sample output
sna::closeness(Simpsons_n, gmode="graph")
[1] 0.5294118 0.5294118 0.6000000 0.5000000 0.5000000
[6] 0.6428571 0.6428571 0.6000000 0.4285714 0.3103448
```

```r
igraph::closeness(graph, vids = V(graph),
                  mode = c("out", "in", "all", "total"),
                  weights = NULL, normalized = FALSE) # default
# Sample output
igraph::closeness(Simpsons_i, normalized = T)
      Ned     Marge     Homer       Abe    Maggie
0.5294118 0.5294118 0.6000000 0.5000000 0.5000000
     Bart      Lisa    Krusty       Bob     Cecil
0.6428571 0.6428571 0.6000000 0.4285714 0.3103448
```

Remark. For the `igraph::closeness()` function, we need to specify `normalized = T`. However, when there exists multiple components (for undirected, non-valued graph), this Freeman Closeness Centrality **no longer works** (Why? every total geodesic steps $= \infty$).



Replication. Figure in LINKS Workshop 2023.

Remark. With tie Bob-to-Cecil, the central nodes are Lisa and Bart. However, without such tie, the new central node becomes Homer.

Motivation. When there exists multiple components, if you have information that you want *as many people as possible* in the group to have, and you can only give it to one person in the group, who would you give it to?

Definition 9.5 (Reciprocal Closeness Centrality). Sum of $\frac{1}{\textit{total geodesic steps}}$ of a given node and divided by the theoretical minimum farness.

Example 9.3 (Sample R codes: Reciprocal)**.** Below are some Reciprocal Closeness Centralities using *sna*. We first set Bob and Cecil unlinked, i.e., more than 1 components.

```
Simpsons_n2 = Simpsons_n
Simpsons_n2[9,10] = 0

#Sample output
sna::closeness(Simpsons_n2, gmode = "graph", cmode = "suminvundir")
[1] 0.6481481 0.6481481 0.7592593 0.5925926 0.5925926
[6] 0.7222222 0.7222222 0.6111111 0.4074074 0.0000000
```

Adversely, we can let Krusty and Bob unlinked (still more than 1 components).

```
Simpsons_n3 = Simpsons_n
Simpsons_n3[8,9] = 0

#Sample output
sna::closeness(Simpsons_n3, gmode="graph", cmode="suminvundir")
[1] 0.6111111 0.6111111 0.7222222 0.5555556 0.5555556
[6] 0.6666667 0.6666667 0.5000000 0.1111111 0.1111111
```

9.3 Betweenness Centrality

Motivation. Who is important as "in-between" person to transfer information?

Definition 9.6 (Betweenness Centrality)**.** (skip)

Example 9.4 (Sample R Code: Betweenness)**.** Below are some Betweenness Centrality using *sna* and *igraph*.

```
sna::betweenness(Simpsons_n, gmode = "graph")
[1] 0.8333333 0.8333333 3.6666667 0.0000000 0.0000000
[6] 8.3333333 8.3333333 14.0000000 8.0000000 0.0000000

igraph::betweenness(Simpsons_i)
      Ned     Marge      Homer        Abe     Maggie
0.8333333 0.8333333  3.6666667 0.0000000 0.0000000
     Bart       Lisa     Krusty        Bob      Cecil
8.3333333 8.3333333 14.0000000 8.0000000 0.0000000
igraph::betweenness(Simpsons_i, normalized=T)
       Ned       Marge      Homer         Abe      Maggie
0.02314815 0.02314815 0.10185185 0.00000000 0.00000000
      Bart        Lisa      Krusty        Bob       Cecil
0.23148148 0.23148148 0.38888889 0.22222222 0.00000000
```

10 Group-level Measures

Overview. Key focuses: density, centralization, reciprocity, transitivity, components.

Definition 10.1 (Density)**.** *Density* is defined as the proportion of pairs of nodes are directly connected in a network.

Example 10.1. For Undirected network with 8 nodes and 7 ties, the density is measured by $\frac{\text{Number of ties}}{\text{Number of possible ties}} = \frac{7}{C_2^8} = 0.25$.

Definition 10.2 (Centralization)**.** aaa

(Last edited: 06/06/2023, Borgatti et al. (2022) & LINKS Workshop 2023)

References

Borgatti, S. P., Everett, M. G., Johnson, J. C., & Agneessens, F. (2022). Analyzing Social Networks Using R (1st ed.). *SAGE Publications*.