HW 2 Report

Yongchao Qiao

Introduction

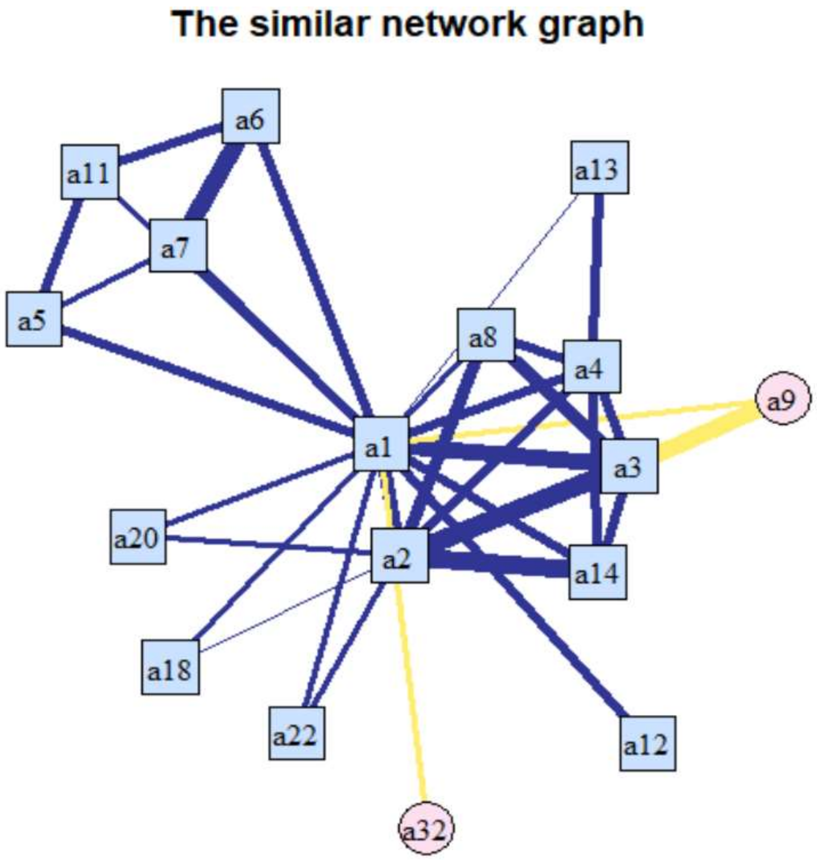
There are two important topics, including plotting network and network graphic design, for the network. In the first topic, the aims for plotting network are to understand overall structure, hopefully to visualize essential information and sometimes to draw conclusions. However, there are too many ways to draw the graph, then network layout design has been applied to illustrate the essential information and reduce irrelevant information of the network. Generally, five conditions should be considered for the layout design, which are minimizing edge crossings, maximizing the symmetry of the layout of nodes, minimizing the variability of the edge lengths, maximizing the angle between edges when they cross or join nodes and minimizing the total space used for the network display. Typically, the force-directed algorithm has been widely used to meet these conditions.

As for network graphic design, the important principle is to reveal important or interesting information that is contained in the network data. The design elements contain nodes' color, nodes' shape, nodes' size, nodes' label, edges' width, edges' color, edges' type and legends. Besides, there are three centralities to adjust these elements. The first one is degree centrality, which is simply the degree of each node. As for closeness centrality, it means that nodes are more prominent to the extent they are close to all other nodes in the network. Finally, betweenness centrality measures the extent that a node sits 'between' pairs of other nodes in the network, such that a path between the other nodes has to go through that node.

Construct the network

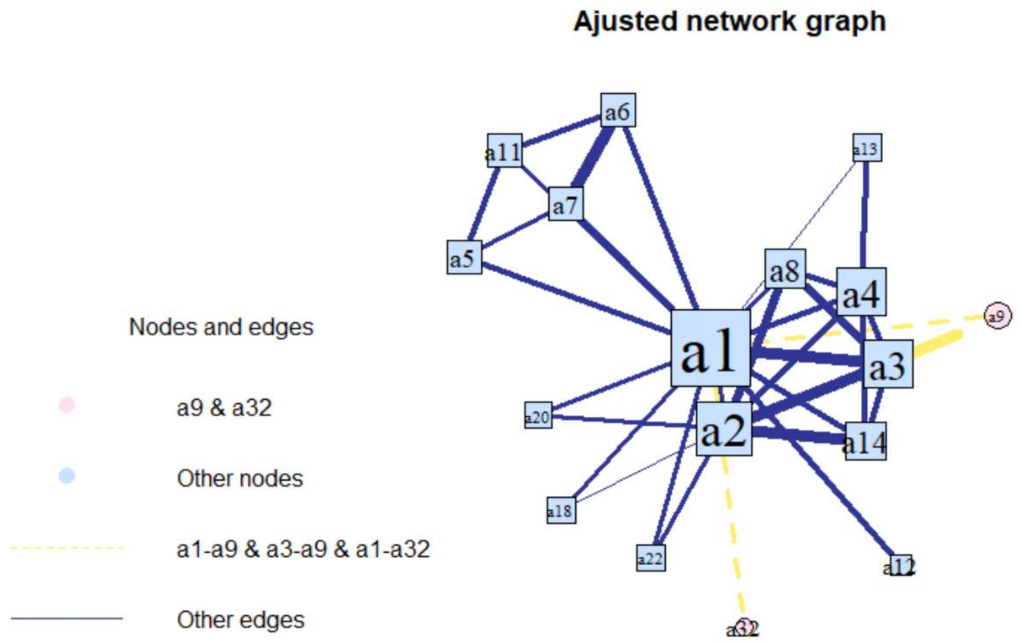
```
##   Network edge list matrix:
##       [,1] [,2]
##  [1,]    1    2
##  [2,]    1    3
##  [3,]    1    4
##  [4,]    1    5
##  [5,]    1    6
##  [6,]    1    7
##  [7,]    1    8
##  [8,]    1    9
##  [9,]    1   10
## [10,]    1   11
## [11,]    1   12
## [12,]    1   13
## [13,]    1   14
## [14,]    1   15
## [15,]    1   16
## [16,]    1   17
## [17,]    2    3
## [18,]    2    4
## [19,]    2    8
## [20,]    2   13
## [21,]    2   14
## [22,]    2   15
## [23,]    2   16
## [24,]    3    4
## [25,]    3    8
## [26,]    3    9
## [27,]    3   13
## [28,]    4    8
## [29,]    4   12
## [30,]    4   13
## [31,]    5    7
## [32,]    5   10
## [33,]    6    7
## [34,]    6   10
```

Create a similar graph



Generally, this graph is similar with the one in the slides, from nodes' shape and color to labels' and edges' color.

Adjust the graph



From the result, we can see that the graph has been improved by degree centrality, different edges types based on different colors and the legend.

Appendix

```
library(UserNetR)
library(statnet)

library(RColorBrewer)
```

Construst a network data

```
netmat = rbind(c(1,2),
               c(1,3),
               c(1,4),
               c(1,5),
               c(1,6),
               c(1,7),
               c(1,8),
               c(1,9),
               c(1,10),
               c(1,11),
               c(1,12),
               c(1,13),
               c(1,14),
               c(1,15),
               c(1,16),
               c(1,17),

               c(2,3),
               c(2,4),
               c(2,8),
               c(2,13),
               c(2,14),
               c(2,15),
               c(2,16),

               c(3,4),
               c(3,8),
               c(3,9),
               c(3,13),

               c(4,8),
               c(4,12),
               c(4,13),

               c(5,7),
               c(5,10),

               c(6,7),
               c(6,10))

net =network(netmat, matrix.type = "edgelist") # Use edgelist to creat the
network

network.vertex.names(net) =c("a1","a2","a3","a4","a5","a6","a7","a8","a9",
"a11","a12","a13","a14","a18","a20","a22","a32") # Set nodes' names

summary(net) # Summary the network

set.vertex.attribute(net,"Color",c("lightsteelblue1","lightsteelblue1","li
ghtsteelblue1","lightsteelblue1","lightsteelblue1","lightsteelblue1","ligh
tsteelblue1","lightsteelblue1",brewer.pal(11,"PiYG")[5],"lightsteelblue1",
"lightsteelblue1","lightsteelblue1","lightsteelblue1","lightsteelblue1","l
ightsteelblue1","lightsteelblue1",brewer.pal(11,"PiYG")[5])) # Create colo
r attribute of nodes

set.edge.attribute(net,"Width",c(3,3.9,2,2,2,2,2,1.5,1.5,1.5,2,0.5,2,1.5,1.
5,1.5,1.5,4.5,2,3,4,0.5,1.5,1.5,2,3,4,2,2,2,2,1.5,2,4,2))  # Create width a
ttribute of edges
```

```
set.edge.attribute(net,"Edge Color", c(1,1,1,1,1,1,1,2,1,1,1,1,1,1,1,2,1,
1,1,1,1,1,1,1,1,2,1,1,1,1,1,1,1,1))  # Create color attribute of edges

deg = sna::degree(net)
set.vertex.attribute(net,"Degree", deg) # Create degree attribute of nodes
summary(net)  # Smuuary the network
```

Create a similar graph

```
detach(package:statnet,unload=TRUE)
library(igraph)

library(intergraph)

my_edge_color = c(brewer.pal(11,"RdYlBu")[11],brewer.pal(12,"Set3")[12])
# Get the colors which are saved as my_edge_color

net1 <- asIgraph(net)  # Transfer the network to igraph
V(net1)$name <- c("a1","a2","a3","a4","a5","a6","a7","a8","a9","a11","a12
","a13","a14","a18","a20","a22","a32") #Set the igraph nodes' name

shapes=c("circle","square") # Set the shape parameters
set.seed(225)
plot(net1,   #Plot the similar graph
    vertex.color =V(net1)$Color, #Set the nodes' colors
    vertex.shape = shapes[as.factor(V(net1)$Color)], #Set the nodes' shape
s
    vertex.size = 15, #Set the nodes' size
    edge.arrow.mode = 0, # Not show the arrows
    vertex.label.color = "black", # Set the labels' color
    edge.color = my_edge_color[E(net1)$Edge.Color], #set the edges' color
    edge.width = E(net1)$Width*2.5, #Set edges' width
    displaylabels = TRUE, # Show the labels
    main = "The similar network graph") #Set the title of the graph
```

Adjust the graph

```
set.seed(225)
plot(net1,  #Plot the adjusted graph
    vertex.color =V(net1)$Color,  #Set the nodes' colors
    vertex.shape = shapes[as.factor(V(net1)$Color)], #Set the nodes' shape
s
    vertex.size = sqrt(V(net1)$Degree*55), #Set the nodes' size
    edge.arrow.mode = 0,  #Not show the arrows
    edge.color = my_edge_color[E(net1)$Edge.Color], # Set edges' colors
    edge.width = E(net1)$Width*2.1,  #Set edges' width
    edge.lty = E(net1)$Edge.Color,   # Set edges' type
    vertex.label.color = "black",    # Set labels' color
    vertex.label.cex = log(V(net1)$Degree),  #Set labels' size
    displaylabels = TRUE,     #Show the labels
    main = "Ajusted network graph")  # Set the title of the graph
legend("bottomleft",  # Set the legend's position
      legend=c("a9 & a32","Other nodes", "a1-a9 & a3-a9 & a1-a32", "Other edges"),   #Set the legend's notes
      col = c(brewer.pal(11,"PiYG")[5],"lightsteelblue1",my_edge_color[2],my_edge_color[1]),   #Set the notations' color
      pch=c(19,19,-1,-1),    # Set the notations' type
      lty = c(-1,-1,2,1),    # Set the notations' type
      pt.cex=1.5,  #Expansion factor(s) for the points
      bty="n", #The type of box to be drawn around the legend.
title="Nodes and edges",   # Set legend's name
title.adj = 0.2)   # Set legend's name's position
```