Assignment 2 - Social Network Analysis

Part I

Start by installing the "igraph" package. Once you have installed igraph, load the package.

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
## The following object is masked from 'package:base':
##
##     union
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:igraph':
##
##     as_data_frame, groups, union
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
## The following object is masked from 'package:igraph':
##
##     crossing
```

Now upload the data file "discipline-data.csv" as a data frame called "D1".

```
D1 <- read.csv("discipline-data.csv")
```

Each row is a disciplinary action from a teacher to a student so the first line shows that teacher "E" sent student "21" to the principal. It also shows the gender of both the teacher and student and the student's main elective field of study ("major""") and the field that the teacher instructs in ("t.expertise").

Before you proceed, you will need to change the data type of the student id variable. Since it is a number R will automatically think it is an integer and code it as such (look at the list of variables by clicking on the data frame arrow in the Data pane. Here you will see the letters "int"" next to the stid variable, that stands for integer). However, in this case we are treating the variable as a category, there is no numeric meaning in the variable. So we need to change the format to be a category, what R calls a "factor". We can do this with the following code:

```
D1$stid <- as.factor(D1$stid)
```

igraph requires data to be in a particular structure. There are several structures that it can use but we will be using a combination of an "edge list" and a "vertex list". As you might imagine the edge list contains a list of all the relationships between students and teachers and any characteristics of those edges that we might be interested in. There are two essential variables in the edge list a "from" variable and a "to" variable that descibe the relationships between vertices (a disciplinary action is given "from" and teacher "to" a student). While the vertix list contains all the characteristics of those vertices, in our case gender and major.

So let's convert our data into an edge list!

First we will isolate the variables that are of interest: tid and stid

```
library(dplyr)

D2 <- dplyr::select(D1, tid, stid)
```

Since our data represnts every time a teacher sends a student to the principal there are multiple rows when the same teacher sends the same student. We want to collapse these into a single row, with a variable that shows how many times a teacher-student pair appears.

```
EDGE <- dplyr::count(D2, tid, stid)

names(EDGE) <- c("from", "to", "count")
```

EDGE is your edge list. Now we need to make the vertex list, a list of all the teachers and students and their characteristics in our network.

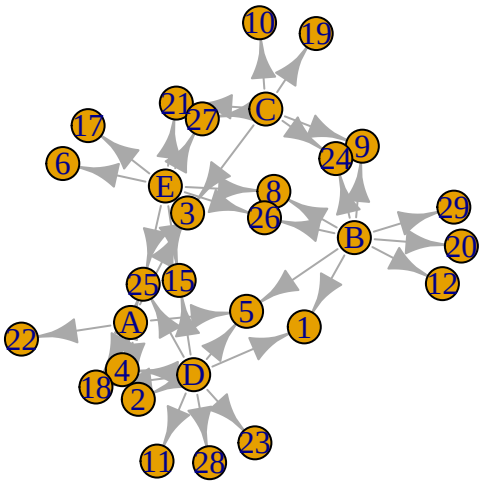
```
#First we will separate the teachers from our original data frame
V.TCH <- dplyr::select(D1, tid, t.gender, t.expertise)
#Remove all the repeats so that we just have a list of each teacher and their characteristics
V.TCH <- unique(V.TCH)
#Add a variable that describes that they are teachers
V.TCH$group <- "teacher"

#Now repeat this process for the students
V.STD <- dplyr::select(D1, stid, s.gender, s.major)
V.STD <- unique(V.STD)
V.STD$group <- "student"

#Make sure that the student and teacher data frames have the same variables names
names(V.TCH) <- c("id", "gender", "topic", "group")
names(V.STD) <- c("id", "gender", "topic", "group")

#Bind the two data frames together (you will get a warning because the teacher data frame has 5 types o
VERTEX <- dplyr::bind_rows(V.TCH, V.STD)
```

```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector

## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector
```

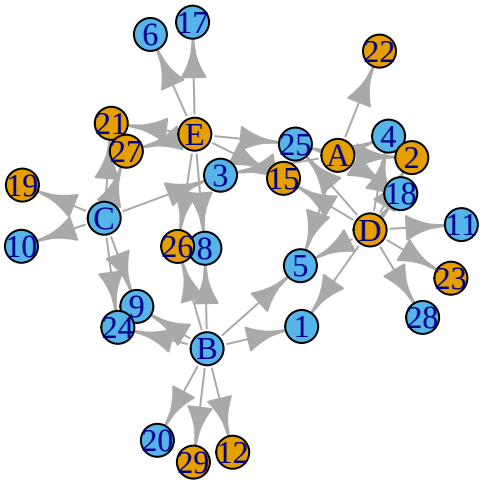
```
## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector
```

Now we have both a Vertex and Edge list it is time to plot our graph!

```
#Load the igraph package

library(igraph)

#First we will make an object that contains the graph information using our two dataframes EDGE and VER

g <- graph.data.frame(EDGE, directed=TRUE, vertices=VERTEX)

#Now we can plot our graph using the force directed graphing technique - our old friend Fruchertman-Rei

plot(g,layout=layout.fruchterman.reingold)
```

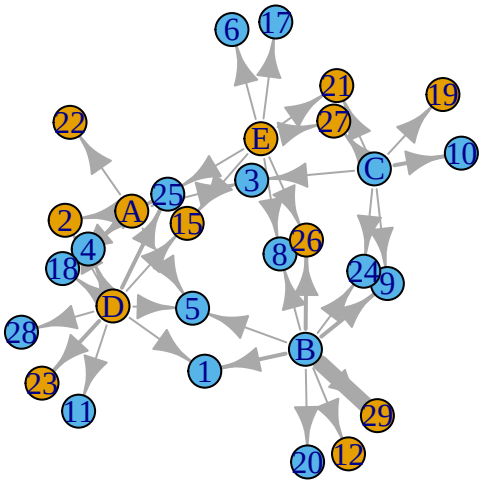



```
#There are many ways to change the attributes of the graph to represent different characteristics of th

plot(g,layout=layout.fruchterman.reingold, vertex.color=VERTEX$gender)
```




```
#We can change the thickness of the edge according to the number of times a particular teacher has sent
```

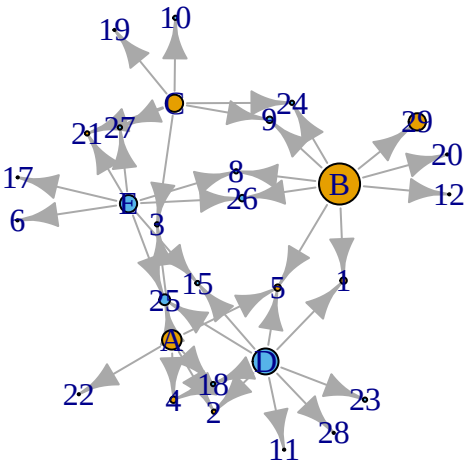
```
plot(g,layout=layout.fruchterman.reingold, vertex.color=VERTEX$gender, edge.width=EDGE$count)
```



Part II

In Part II your task is to look up in the igraph documentation and create a graph that sizes the student vertices in terms of the number of disciplinary actions they have recieved, and the teachers in terms of the number of disciplinary actions they have given out.

```
t.size <- EDGE %>%
  group_by(from) %>%
  summarise(sum(count))
names(t.size) <- c("id", "size")
s.size <- EDGE %>%
  group_by(to) %>%
  summarise(sum(count))
names(s.size) <- c("id", "size")
VERTEX2 <- merge(VERTEX, t.size, by="id")
VERTEX3 <- merge(VERTEX, s.size, by="id")
VERTEX4 <- bind_rows(VERTEX2, VERTEX3)

g2 <- graph.data.frame(EDGE, directed=TRUE, vertices=VERTEX4)
plot(g2,layout=layout.fruchterman.reingold, vertex.color=VERTEX$gender, vertex.size = VERTEX4$count)
```



Part III

Now practice with data from our class. Please create a **person-network** with the data set hudk4050-classes.csv.

```
Data1 <- read.csv("hudk4050-classes.csv")
```

To create this network you will need to create a person-class matrix using the tidyr functions and then create a person-person matrix using `t()`.

```
Data2 <- unite(Data1, Name, "Last.name", "First.name")
Data3 <- gather(Data2, Course, class, "Course1", "Course2", "Course3", "Course4", "Course5", convert = I
```

```
## Warning: attributes are not identical across measure variables;
## they will be dropped
```

```
Data4 <- select(Data3, Name, class)
Data5 <- filter(Data4, class != "HUDK4050")
Data5$n <- 1
Data5 <- filter(Data5, class>0)
Data6<-spread(Data5,class, n)
row.names(Data6) <- Data6$Name
Data6$Name <- NULL
Data6 <- ifelse(is.na(Data6),0,1)
Data7 <- as.matrix(Data6)
Data8 <-Data7%*%t(Data7)
diag(Data8) <- NA
```

You will then need to plot a matrix rather than a data frame using igraph.

```
g2 <- graph.adjacency(Data8)
plot(g2,layout=layout.fruchterman.reingold)
```

Once you have done this, also look up how to generate the following network metrics: betweeness centrality and dregree. **Who is the most central person in the network?**

```
which.max(degree(g2))
```

```
## Cody_David
##          2
```

```
which.max(betweenness(g2))
```

```
## Bennett Colomer_Magdalena
##                         1
```

```
plot(g2,layout=layout.fruchterman.reingold,vertex.size=degree(g2)/5,edge.width=edge_betweenness(g2)/10)
```