Assignment 1

Smitesh Patil

2023-02-07

Section 1. Basic measures

```
library(igraph)
library(kableExtra)
library(gt)
dib_graph<-read.graph("dib2.graphml",format="graphml")
```

1.1 Give the number of nodes and edges

```
cat("num vertices:", vcount(dib_graph), "\n")
```

```
## num vertices: 8969
```

```
cat("num edges :", ecount(dib_graph), "\n")
```

```
## num edges : 46750
```

Is the network strongly or weakly connected. If neither, what is the distribution of component sizes.

```
strong_component = as.data.frame(table(factor(components(dib_graph, mode="strong")$csize)))
names(strong_component)[1] = "Component Size"

strong_component %>% gt() %>%
  tab_header("Strong Number of Components by size of the Compenent")
```

1. Strongly connected components

Strong Number of Components by size of the Compenent

Component Size	Freq
1	3024
2	180
3	25
4	5
5	1
6	1
5479	1

Answer: The table above shows strongly connected components in the directed network. There is one component with 5479, 6 and 5 nodes and a distribution of component with sizes varying from 1 to 4.

2. Weakly connected components

```
weak_component = as.data.frame(table(factor(components(dib_graph, mode="weak")$csize)))
names(weak_component)[1] = "Component Size"

weak_component %>% gt() %>%
  tab_header("Weak Number of Components by size of
    the Compenent")
```

Weak Number of Components by size of the Compenent

Component Size	Freq
2	30
3	11
4	1
8872	1

Answer: The table above shows weakly connected components in the directed network. There is one component with 8872 and 4 nodes with 30 weakly connected components of size 2 and 11 components with size 3.

1.3 What is the diameter of the network ?

```
cat("The diameter of the network is : ", diameter(dib_graph, directed = T,
    unconnected = TRUE, weights = NA), "\n")
```

```
## The diameter of the network is :  18
```

1.4 What is the average path length of the network ?

```
cat("The average path length of the network :", mean_distance(dib_graph,
    directed = T), "\n")
```

```
## The average path length of the network : 6.017593
```

1.5 What is the clustering coefficient of the network ?

```
cat("The clustering coeff of the graph is :", transitivity(dib_graph, type = "localaverage"),
    "\n")
```

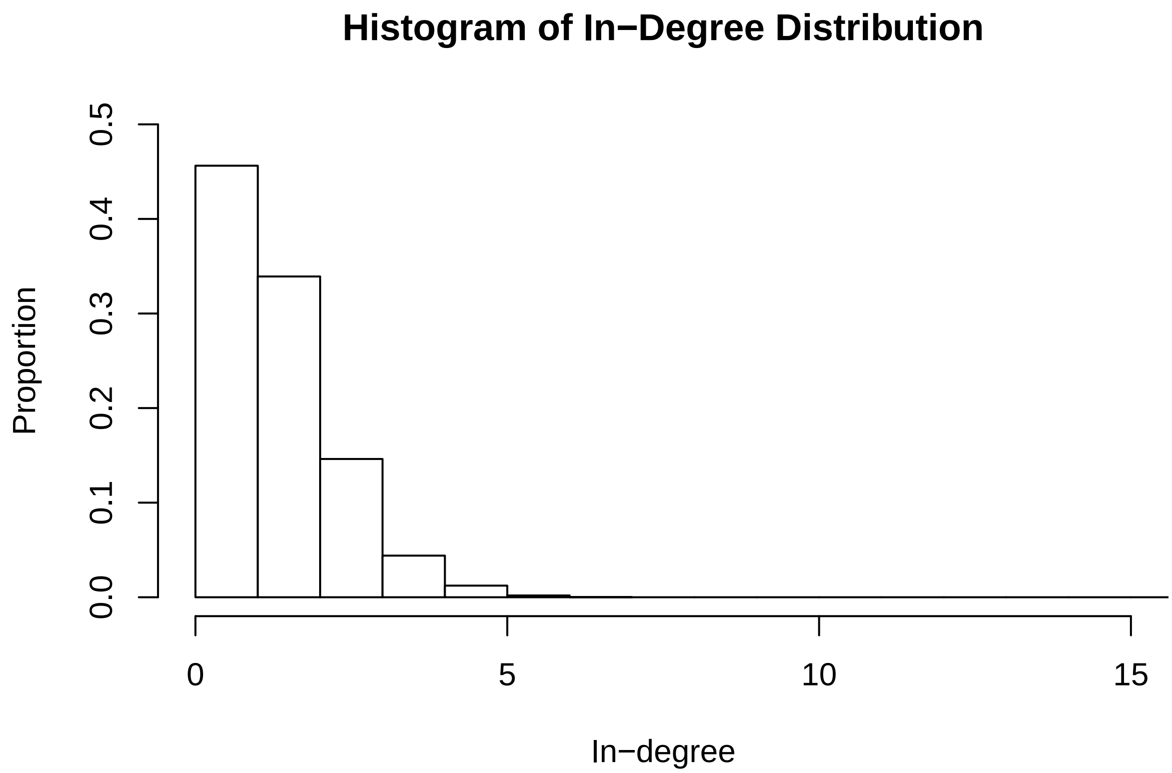
```
## The clustering coeff of the graph is : 0.2300017
```

1.6 What is the in- and out-degree distribution ?

```
deg <- degree(dib_graph, mode = "in")
cat("The in-degree distribution of the graph varies from ", min(deg), "to ",
    max(deg))
```

```
## The in-degree distribution of the graph varies from  0 to  473
```

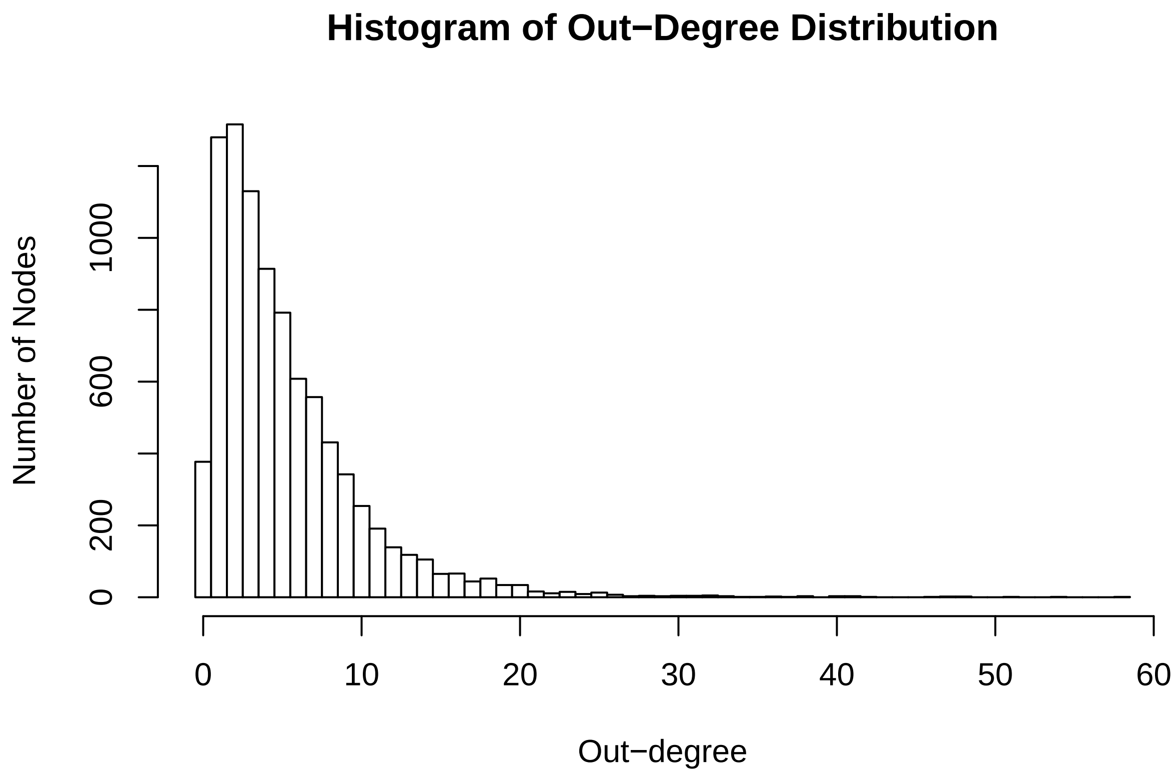
```
hist(log(as.data.frame(deg)$deg), breaks = (min(deg)):(max(deg)), xlab = "In-degree",
    freq = FALSE,
    ylab = "Proportion", main = "Histogram of In-Degree Distribution",
    border = "black", col = "white",
    xlim = c(0,15),
    ylim = c(0,0.5))
```




```r
deg <- degree(dib_graph, mode = "out")
cat("The in-degree distribution of the graph varies from ", min(deg), "to ",
    max(deg))
```

```
## The in-degree distribution of the graph varies from  0 to  58
```

```r
hist(deg, breaks = (min(deg) - 1):(max(deg)) + 0.5, xlab = "Out-degree",
    ylab = "Number of Nodes", main = "Histogram of Out-Degree Distribution",
    border = "black", col = "white")
```



Section 2 - Bowtie Analysis

```
## Strongly connected components

#get the cluster in strongly connected graph with max
#size and find the rest of nodes connected thats SCC
cluster <- components(dib_graph, mode = "strong")

scc_index <- which.max(cluster$csize)
scc <- V(dib_graph)[which(cluster$membership == scc_index)]$name

# get the career and century for future analysis
scc_career <- V(dib_graph)[which(cluster$membership == scc_index)]$career
scc_century <- V(dib_graph)[which(cluster$membership == scc_index)]$century


# IN components

#Nodes connected to scc in OUT mode from the scc
# get the career and century for future analysis

IN_component = c()
IN_component_career = c()
```

```r
IN_component_century = c()
vertices <- V(dib_graph)

non_SCC <- vertices[!(vertices$name %in% scc)]

for (v in non_SCC) {

    dist <- bfs(dib_graph, root = v, mode = "out", unreachable = F, dist = T)$dist

    connected_to_SCC <- !is.nan(dist[scc])

    if (TRUE %in% connected_to_SCC) {
        IN_component <- c(IN_component, V(dib_graph)[v]$name)
        IN_component_career <- c(IN_component_career, V(dib_graph)[v]$career)
        IN_component_century <- c(IN_component_century, V(dib_graph)[v]$century)
    }
}

# OUT components

#Nodes connected to scc in IN mode from the scc
# get the career and century for future analysis

OUT_component = c()
OUT_component_career = c()
OUT_component_century = c()

non_SCC2 <- vertices[!(vertices$name %in% c(scc, IN_component))]

for (v in non_SCC2) {
    dist <- bfs(dib_graph, root = v, mode = "in", unreachable = F, dist = T)$dist
    connected_to_non_SCC <- !is.nan(dist[scc])
    if (TRUE %in% connected_to_non_SCC) {
        OUT_component <- c(OUT_component, V(dib_graph)[v]$name)
        OUT_component_career <- c(OUT_component_career, V(dib_graph)[v]$career)
        OUT_component_century <- c(OUT_component_century, V(dib_graph)[v]$century)
    }
}

# Tube components

# logic : from vertices connected to the in component check the
# vertices connected to the put component in out mode

# get the career and century for future analysis

tube = c()
tube_career = c()
tube_century = c()
nodes_minus_SCC <- vertices[!(vertices$name %in% scc)]
g_minus_SCC <- induced_subgraph(dib_graph, nodes_minus_SCC)
out = c()
```

Table 3: Network Composition

	Percentage
SCC	0.6108819
IN Component	0.3075036
OUT component	0.0248634
Tube	0.0013379

```
for (v in V(g_minus_SCC)[IN_component]) {
    paths <- all_simple_paths(g_minus_SCC, from = v, to = V(g_minus_SCC)[OUT_component],
        mode = "out", cutoff = -1)
    tube <- c(tube, names(unlist(paths)))
}

tube <- unique(tube)

tube <- tube[!(tube %in% c(IN_component, OUT_component))]


tube_data = c()
tube_data <- V(dib_graph)[V(dib_graph)$name %in% tube]
tube_data_career <- tube_data$career
tube_data_century <- tube_data$century
```

2.1 What percentage of the network comprises the strongly connected component, the incomponent, the out-component, and tube.

```
library(dplyr)

# percentages void of tendrils and disconnected vertices

components_prop <- c(length(scc)/length(vertices), length(IN_component)/length(vertices),
    length(OUT_component)/length(vertices), length(tube)/(length(vertices)))

names(components_prop) <- c("SCC", "IN Component", "OUT component", "Tube")

kable(components_prop, col.names = "Percentage", caption = "Network Composition")
```

2.2 What are the top five careers for people in the in-component, out-component, scc and tube components?

```
library(dplyr)

#list of careers splited by the comma and combined in a single list
#the  sort by descending order and take top 5
```

```r
scc_career <- trimws(unlist(strsplit(scc_career, split = ",")))
scc_career <- as_tibble(table(scc_career)) %>%
    arrange(desc(n)) %>%
    head(5)


IN_component_career <- trimws(unlist(strsplit(IN_component_career, split = ",")))
IN_component_career <- as_tibble(table(IN_component_career)) %>%
    arrange(desc(n)) %>%
    head(5)

OUT_component_career <- trimws(unlist(strsplit(OUT_component_career, split = ",")))
OUT_component_career <- as_tibble(table(OUT_component_career)) %>%
    arrange(desc(n)) %>%
    head(5)


tube_component_career <- trimws(unlist(strsplit(tube_data_career, split = ",")))
tube_component_career <- as_tibble(table(tube_component_career)) %>%
    arrange(desc(n)) %>%
    head(5)

names(scc_career)[1] <- "SCC career"
names(IN_component_career)[1] <- "In career"
names(OUT_component_career)[1] <- "Out career"
names(tube_component_career)[1] <- "Tube career"

names(scc_career)[2] <- "SCC Count"
names(IN_component_career)[2] <- "In Count"
names(OUT_component_career)[2] <- "Out Count"
names(tube_component_career)[2] <- "Tube Count"

first_table <- cbind(scc_career, IN_component_career) %>%
    gt() %>%
    tab_header("Top Five Career in SCC and IN Component") %>%
    tab_spanner(label = "SCC", columns = c("SCC career", "SCC Count")) %>%
    tab_spanner(label = "IN Component", columns = c("In career", "In Count"))

second_table <- cbind(OUT_component_career, tube_component_career) %>%
    gt() %>%
    tab_header("Top Five Career in OUT Component and Tube") %>%
    tab_spanner(label = "OUT Component", columns = c("Out career", "Out Count")) %>%
    tab_spanner(label = "Tube Component", columns = c("Tube career", "Tube Count"))
first_table
```

Top Five Career in SCC and IN Component

SCC		IN Component	
SCC career	SCC Count	In career	In Count
Politics	1909	Politics	591
Religion	1004	Religion	487
Literature	587	Literature	312

Military	502	Business and Finance	240
Gentry and Aristocracy	486	Journalism and Broadcasting	224

```
second_table
```

Top Five Career in OUT Component and Tube

OUT Component		Tube Component	
Out career	Out Count	Tube career	Tube Count
Religion	37	Military	3
Science and Technology	26	Science and Technology	2
Politics	25	The Sea	2
Sport	20	Travel and Exploration	2
Administration and Diplomacy	18	Administration and Diplomacy	1

2.3 What are the top five centuries represented in the in-component, out-component, scc and tube components?

Code

```
library(dplyr)

#list of centuries in single list the  sort by descending
#order and take top 5

scc_century <- as_tibble(table(scc_century)) %>%
    arrange(desc(n)) %>%
    head() %>%
    mutate(scc_century = paste(scc_century , "century"))

IN_component_century <- as_tibble(table(IN_component_century)) %>%
    arrange(desc(n)) %>%
    head() %>%
    mutate(IN_component_century =  paste(IN_component_century, "century"))

OUT_component_century <- as_tibble(table(OUT_component_century)) %>%
    arrange(desc(n)) %>%
    head() %>%
    mutate(OUT_component_century = paste(OUT_component_century, "century"))

tube_component_century <- as_tibble(table(tube_data_century)) %>%
    arrange(desc(n)) %>%
    head() %>%
    mutate(tube_data_century = paste(tube_data_century, "century"))

names(scc_century)[1] <- "SCC century"
```

```r
names(IN_component_century)[1] <- "In century"
names(OUT_component_century)[1] <- "Out century"
names(tube_component_century)[1] <- "Tube century"

names(scc_century)[2] <- "SCC count"
names(IN_component_century)[2] <- "In count"
names(OUT_component_century)[2] <- "Out count"
names(tube_component_century)[2] <- "Tube count"


cbind(scc_century, IN_component_century, OUT_component_century, tube_component_century) %>%
    gt() %>%
    tab_header("Top Five Century in SCC, IN Component, OUT Component, Tube") %>%
    tab_spanner(label = "SCC", columns = c("SCC century", "SCC count")) %>%
    tab_spanner(label = "IN Component", columns = c("In century", "In count")) %>%
    tab_spanner(label = "OUT Component", columns = c("Out century", "Out count")) %>%
    tab_spanner(label = "Tube Component", columns = c("Tube century", "Tube count"))
```

Top Five Century in SCC, IN Component, OUT Component, Tube

SCC		IN Component		OUT Component		Tube Component	
SCC century	SCC count	In century	In count	Out century	Out count	Tube century	Tube count
19 century	1871	19 century	1090	19 century	112	20 century	8
18 century	1133	20 century	789	20 century	52	19 century	3
20 century	597	18 century	419	18 century	39	18 century	1
17 century	595	17 century	170	17 century	10	20 century	8
16 century	474	16 century	120	16 century	4	19 century	3
15 century	118	13 century	36	13 century	3	18 century	1

2.5 Write R code to detect whether there are tendrils connecting to the in-component and tendrils connecting from the out-component. Show that your code is working correctly and can find tendrils accurately where they are present in a network.

```r
### Writing code to detect in and out tendril and check it initally on week 2
### lab data then apply the code on the dib graph


### Tendril detection on graph from week2 lab

# week 2 code

scc_comp <- c("Maria","John", "Sue", "Maria", "Sue", "John",
              "Zoe","Sue", "John", "Zoe","John","Paul", "Paul","Zoe")

# out-component
out_comp <- c("Peter", "Petra", "Petra", "Greta", "Petra", "Ina",
              "Petra","Carlos", "Carlos","Owen","Carlos","Noel",
              "Carlos","Mark")

# in -component
```

```r
in_comp <- c("Jane", "Rose", "Rose", "Sean", "Rose", "Steve", "Sean","Steve", "Sean","Karen")

# tube component
tube_comp<- c("Una","Vera","Vera","Con", "Una","Nora", "Nora","Con")

# isolated component

iso1 <-c("Tobias", "Shirvo","Shirvo", "Lucy", "Lucy", "Tobias", "Lucy", "Pam")

iso2 <-c("Ron", "Joey","Joey", "Tadhg", "Ron", "Ken","Ken", "Mags")


# create a vector of edges between the components

# from in_component to scc
in_scc<- c("Steve","Maria", "Karen","Sue")
# from scc to out_component
scc_out<- c("Paul","Peter", "Paul","Greta")
# from in_component to tube
in_tube<-c("Sean","Una")
# from tube to out_component
tube_out<-c("Con","Owen")

# make the graph based on the component vertices
g<-make_graph(c(scc_comp, out_comp, in_comp, tube, iso1, iso2),directed=T)


# adding a in-tendril Jane -> Smitesh Patil
g <- g + vertices("Smitesh Patil")
g <- g + edges("Jane", "Smitesh Patil")

# adding a out-tendril Patil Smitesh -> Peter
g <- g + vertices("Patil Smitesh")
g <- g + edges("Patil Smitesh", "Peter")

vertices <- V(g)

# pos_tendril(possible tendril) containes vertices that are not in scc,
#in, out, tube can contain disconnected vertices
in_tendril = c()
pos_tendril = vertices[!(vertices$name %in% c(scc_comp, in_comp, out_comp,
    tube_comp))]$name


# create a subgraph that contains just the In-component and possible
#tendril

nodes_IN_component_Tendrils <- vertices[vertices$name %in% c(pos_tendril, in_comp)]
g_IN_component_Tendrils <- induced.subgraph(g, nodes_IN_component_Tendrils)


# check the nodes in the subgrah from in component to the possible
# tendril
```

```r
for (v in pos_tendril) {
    paths <- all_simple_paths(g_IN_component_Tendrils, from = v,
             to = V(g_IN_component_Tendrils)[in_comp],
             mode = "in", cutoff = -1)

    if (length(paths) > 0) {
        in_tendril <- c(in_tendril, v)
    }
}

# check the nodes in the subgrah from possible tendril to the out
# component

out_tendril <- c()

nodes_OUT_component_Tendrils <- vertices[vertices$name %in% c(pos_tendril,
    out_comp)]

g_OUT_component_Tendrils <- induced.subgraph(g, nodes_OUT_component_Tendrils)
for (v in pos_tendril) {
    paths <- all_simple_paths(g_OUT_component_Tendrils, from = v,
             to = V(g_OUT_component_Tendrils)[out_comp],
             mode = "out", cutoff = -1)
    if (length(paths) > 0) {
        out_tendril <- c(out_tendril, v)
    }
}

cat("IN Tendril detected ", in_tendril)
```

```
## IN Tendril detected  Smitesh Patil
```

```r
cat("OUT Tendril detected ",out_tendril)
```

```
## OUT Tendril detected  Patil Smitesh
```

```r
### tendril detection from code above for dib_graph

vertices = V(dib_graph)

in_tendril = c()
Tendril = vertices[!(vertices$name %in% c(scc, IN_component, OUT_component,
    tube))]$name


nodes_IN_component_Tendrils <- vertices[vertices$name %in% c(Tendril, IN_component)]
g_IN_component_Tendrils <- induced.subgraph(dib_graph, nodes_IN_component_Tendrils)


for (v in Tendril) {
    paths <- all_simple_paths(g_IN_component_Tendrils, from = v,
             to = V(g_IN_component_Tendrils)[IN_component],
```

```r
            mode = "in", cutoff = -1)

    if (length(paths) > 0) {
        in_tendril <- c(in_tendril, v)
    }
}

out_tendril <- c()
nodes_OUT_component_Tendrils <- vertices[vertices$name %in% c(Tendril,
    OUT_component)]

g_OUT_component_Tendrils <- induced.subgraph(dib_graph, nodes_OUT_component_Tendrils)

for (v in Tendril) {
    paths <- all_simple_paths(g_OUT_component_Tendrils, from = v,
            to = V(g_OUT_component_Tendrils)[OUT_component],
            mode = "out", cutoff = -1)

    if (length(paths) > 0) {
        out_tendril <- c(out_tendril, v)
    }
}

cat("There are ", length(in_tendril), " vertices in IN Tendrils")
```

```
## There are  162  vertices in IN Tendrils
```

```r
cat("There are ", length(out_tendril), " vertices in OUT Tendrils")
```

```
## There are  27  vertices in OUT Tendrils
```

Section 3 - Centrality /Authority

3.1 Produce a table that shows the most influential people in each century using 3 different measures of centrality/authority

```r
library(sjmisc)

century_data <- list(
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 1]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 4]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 5]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 6]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 7]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 8]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 9]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 10]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 11]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 12]),
```

```r
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 13]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 14]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 15]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 16]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 17]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 18]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 19]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 20]),
  induced.subgraph(dib_graph, V(dib_graph)[V(dib_graph)$century == 21]))

page_rank <- vector(mode = "list", length = 19)
authority <- vector(mode = "list", length = 19)
eigen_centrality <- vector(mode = "list", length = 19)
between <- vector(mode = "list", length = 19)
close <- vector(mode = "list", length = 19)
century_list <- c("1st century", "4th century", "5th century", "6th century",
                  "7th century", "8th century", "9th century", "10th century",
                  "11th century", "12th century", "13th century", "14th century",
                  "15th century", "16th century", "17th century","18th century",
names(century_data) <- century_list


for (i in 1:length(century_data)) {
    page_rank[[i]] <- page_rank(century_data[[i]], directed = TRUE, damping = 0.85)$vector
    page_rank[[i]] <- page_rank[[i]] %>%
        sort(decreasing = TRUE) %>%
        head(1)
    page_rank[[i]] <- paste(names(page_rank[[i]]))

    eigen_centrality[[i]] <- eigen_centrality(century_data[[i]], weights = NA)$vector
    eigen_centrality[[i]] <- eigen_centrality[[i]] %>%
        sort(decreasing = TRUE) %>%
        head(1)
    eigen_centrality[[i]] <- paste(names(eigen_centrality[[i]]))

    between[[i]] <- betweenness(century_data[[i]], weights = NA)
    between[[i]] <- between[[i]] %>%
        sort(decreasing = TRUE) %>%
        head(1)
    between[[i]] <- paste(names(between[[i]]))

    authority[[i]] <- authority_score(century_data[[i]], scale = TRUE)$vector
    authority[[i]] <- authority[[i]] %>%
        sort(decreasing = TRUE) %>%
        head(1)
    authority[[i]] <- paste(names(authority[[i]]))
}


out <- tibble("Authority" = unlist(authority),
              "Page Rank" = unlist(page_rank),
              "Betweeness" = unlist(between))
out <- rotate_df(out)
```

```r
colnames(out) <- century_list

out <- as.data.frame(out)

first <- out[c("1st century", "4th century", "5th century", "6th century")] %>%
    gt(rownames_to_stub = TRUE) %>%
    tab_header(title = "1. Centrality for centuries from 1st to 6th")

second <- out[c("7th century", "8th century", "9th century", "10th century")] %>%
    gt(rownames_to_stub = TRUE) %>%
    tab_header(title = "2. Centrality for centuries from 7th to 10th")

third <- out[c("11th century", "12th century", "13th century", "14th century")] %>%
    gt(rownames_to_stub = TRUE) %>%
    tab_header(title = "3. Centrality for centuries from 11th to 14th")

fourth <- out[c("15th century", "16th century", "17th century", "18th century")] %>%
    gt(rownames_to_stub = TRUE) %>%
    tab_header(title = "4. Centrality for centuries from 15th to 18th")
fiveth <- out[c("19th century", "20th century", "21st century")] %>%
    gt(rownames_to_stub = TRUE) %>%
    tab_header(title = "5. Centrality for centuries from 19th to 21st")

first
```

1. Centrality for centuries from 1st to 6th

	1st century	4th century	5th century	6th century
Authority	CathaÃr MÃ¡r	LÃ³egaire	St Patrick	St Colmcille
Page Rank	CathaÃr MÃ¡r	Benignus	St Brigit	St Colmcille
Betweeness	CathaÃr MÃ¡r	LÃ³egaire	St Patrick	Ã edÃ¡n

```r
second
```

2. Centrality for centuries from 7th to 10th

	7th century	8th century	9th century	10th century
Authority	AdomnÃ¡n	Donnchad Midi	Flann Sinna	Brian Boru
Page Rank	AdomnÃ¡n	Dublittir	Flann Sinna	Brian Boru
Betweeness	Cellach Cualann	Fedelmid	Flann Sinna	Brian Boru

```r
third
```

3. Centrality for centuries from 11th to 14th

	11th century	12th century	13th century	14th century
Authority	Muirchertach Ua Briain	Henry II	Richard Burgh	Richard II
Page Rank	Muirchertach Ua Briain	John (King of England)	Richard Burgh	Richard II

Betweeness	Muirchertach Ua Briain	RuaidrÃ Ua Conchobair	Richard Burgh	Brian Sreamach O'Brien

```
fourth
```

4. Centrality for centuries from 15th to 18th

	15th century	16th century	17th century	18th century
Authority	Gerald FitzGerald	Hugh O'Neill	James II and VII	Daniel O'Connell
Page Rank	Gerald FitzGerald	Hugh O'Neill	James Butler	Daniel O'Connell
Betweeness	Gerald FitzGerald	Hugh O'Neill	Richard Talbot	Wolfe Tone

```
fiveth
```

5. Centrality for centuries from 19th to 21st

	19th century	20th century	21st century
Authority	Ã‰amon De Valera	Jack Lynch	William James Arlow
Page Rank	Charles Stewart Parnell	Jack Lynch	William James Arlow
Betweeness	Ã‰amon De Valera	Garret FitzGerald	William James Arlow

3.2 Explain what data you have used to determine influence in each century.

Answer:
First, subgraphs were created for each century so that the people from other centuries couldn't influence the centrality scores for the second century.
Second, the connections (directed edges) are used to determine the influence of people in each century.

3.3 Say why the centrality measures you have used are appropriate. Your table should look something like the one shown below. You are expected to create it computationally using a dataframe or tibble to hold the data and a table library such as kableExtra to render the table.

Answer :

1. Authority Centrality : Authority centrality measures the importance of a node/person depending on the number of nodes/people pointing towards it (i.e. the in-degree of node). Basic idea is that if many nodes point to a node the it is likely to be an important node.
2. Page Rank Centrality : As opposed to authority centrality Page Rank centrality ranks the importance of nodes based on its connection to other nodes bases on their importance. Page Rank is an iterative algorithm that ranks all nodes during each iteration and coverges to a optimal solution after a set no of iterations.
3. Betweenness Centraility : Betweenness measures the number of times a node lies in the shortest path between other nodes of the network. Basically, if a person lies as a common link between other peoples connection. He/she has got to be a influential person.