

Appendix Team 3 Quantum_Computing

Group 3

2022-06-26

This R Markdown file will be provided as appendix to the final report of Team 3. Referring this appendix will follow the format of (Appendix I-“outline item of Markdown file”). For example, referring the code of Hypothesis 2 will display as following in the report: (Appendix I-4.H2). Please use a browser to interpret the html file. Otherwise, an attached PDF is available for use, although this was created using the native latex version of RStudio and the plots are not interactive.

0. Install & import packages

Install

```
required_packages <- c("tibble", "dplyr", "tidyr", "tidyverse", "igraph", "qgraph", "stringr", "ggplot2", "visNetwork", "data.table", "centiserve", "reshape2", "remotes", "ggcorrplot")

to_install_packages <- required_packages[!(required_packages %in% installed.packages()[, "Package"])]

if(length(to_install_packages)) invisible(install.packages(to_install_packages, dependencies = TRUE))

remotes::install_github("clementviolet/omnivor")
```

Import

```
required_packages <- c("tibble", "dplyr", "tidyr", "tidyverse", "igraph", "qgraph", "stringr", "ggplot2", "visNetwork", "data.table", "centiserve", "reshape2", "ggcorrplot")

invisible(lapply(required_packages, require, character.only = T))
```

1. Data collection & cleaning

Data set is given and contains patents regarding quantum computing.

1.1 Import data

```
df_qc <- read.csv("Quantum_computing.csv", sep=";")
```

1.2 Clean Data

```
# Drop Rows where Inventors, Applicants or IPC classes or empty or NA
df_qc <- df_qc[!(is.na(df_qc$IPC) | df_qc$IPC==""), ]
df_qc <- df_qc[!(is.na(df_qc$Inventors) | df_qc$Inventors==""), ]
df_qc <- df_qc[!(is.na(df_qc$Applicants) | df_qc$Applicants==""), ]

# df_qc gets transformed where each row represents an inventor
df_qc <- df_qc %>%
  mutate(Inventors=strsplit(Inventors, "\n")) %>%
  unnest(Inventors)

# Clean Inventors & Applicants: Needed to adapt the igraph object cleaning
df_qc$Inventors <- gsub("_", "-", df_qc$Inventors)
df_qc$Inventors <- gsub("-", ".", df_qc$Inventors)
df_qc$Inventors <- chartr("[", "..", df_qc$Inventors)
df_qc$Inventors <- chartr("<", "..", df_qc$Inventors)
df_qc$Applicants <- gsub("_", "-", df_qc$Applicants)
df_qc$Applicants <- sub("_$", "", df_qc$Applicants)
df_qc$Applicants <- gsub("-", ".", df_qc$Applicants)
df_qc$Applicants <- chartr("[", "..", df_qc$Applicants)
df_qc$Applicants <- chartr("<", "..", df_qc$Applicants)

# Replace _ if its last character of cell
df_qc$Inventors <- sub("_$", "", df_qc$Inventors)

# df_qc gets transformed where each row represents an applicant in connection
# with each inventor
df_applicants_qc <- df_qc %>%
  mutate(Applicants=strsplit(Applicants, "\n")) %>%
  unnest(Applicants)
```

2. Create necessary objects

2.1 Create the edge list

```
edge_list <- df_qc %>%
  select(Inventors, Title, Applicants, No) %>%
  inner_join(., select(., Inventors, No), by="No") %>%
  filter(Inventors.x != Inventors.y) %>%
  unique %>%
  arrange(Title, No)

# Include the patents, where there is only one author
for (i in 1:nrow(df_qc)){
  row <- df_qc[i,]
```

```

    if (sum(df_qc$No == row$No) == 1){
      edge_list[nrow(edge_list) + 1,] <- list(row$Inventors, row$Title, row$
        Applicants, row$No, row$Inventors)
    }
  }

#rename to columns
names(edge_list)[names(edge_list) == "Inventors.x"] <- "from"
names(edge_list)[names(edge_list) == "Inventors.y"] <- "to"

# Selecting of applicant type is based on the majority of type occurrences
identify_applicant_type <- function(x){
  industrial <- "_INC_|INC_|LLC_|LLC_|_CORP_|_CORPORATION_"
  academic <- "UNIV_|_UNIV_|INST_|_INST_|_INST|RESEARCH_|_RESEARCH_|_RESEARCH|
    COUNCIL_|_COUNCIL_|_COUNCIL_"
  if (grepl(industrial, x)){
    return("industrial")
  } else {
    if (grepl(academic, x)){
      return("academic")
    } else {
      return("private")
    }
  }
}

edge_list$group <- NA

for (i in 1:nrow(edge_list)){
  row <- edge_list[i,]
  edge_list[i, "group"] <- identify_applicant_type(row$Applicants)
}

```

2.2 Create the edge matrix

```

unique_nodes <- unique(c(edge_list$from, edge_list$to))
edges_length <- length(unique_nodes)

edge_matrix <- matrix(, nrow = edges_length, ncol = edges_length, dimnames =
  list(unique_nodes, unique_nodes))

edge_matrix[is.na(edge_matrix)] = 0

for (i in 1:edges_length) {
  prim_inventor <- edge_list[[i, "from"]]
  sec_inventor <- edge_list[[i, "to"]]
  edge_matrix[prim_inventor, sec_inventor] <- edge_matrix[prim_inventor, sec_
    inventor] + 1
}

rm(list=c("prim_inventor", "sec_inventor", "i"))

```

2.3 Create the graph object

```
df_visNetwork <- data.frame(from=edge_list$from, to=edge_list$to)
network_graph <- graph_from_data_frame(
  df_visNetwork,
  directed = FALSE) %>%
  simplify(remove.loops = TRUE)
```

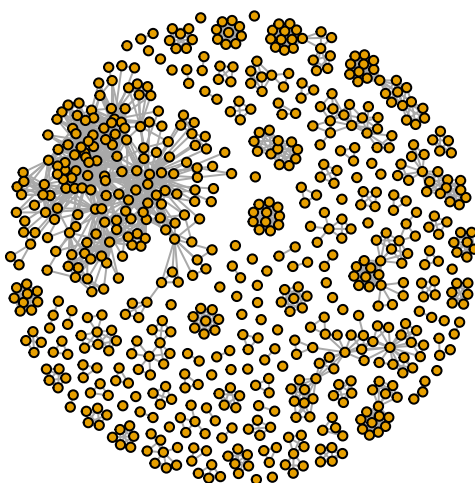
3. Network visualization

3.1 Preview of network graph with fruchtermanreingold Layout

```
# Use qgraph to plot large graphs
edge_list_from_igraph <- get.edgelist(network_graph, names=FALSE)

qgraph_layout_modified <- qgraph.layout.fruchtermanreingold(edge_list_from_igraph,
  vcount=vcount(network_graph), area=8*(vcount(network_graph)^2),
  repulse.rad=(vcount(network_graph)^3.1))

plot(network_graph, layout=qgraph_layout_modified, vertex.size=4, vertex.label=NA)
mtext("Network_graph_based_on_fruchtermanreingold_layout", side=1)
```

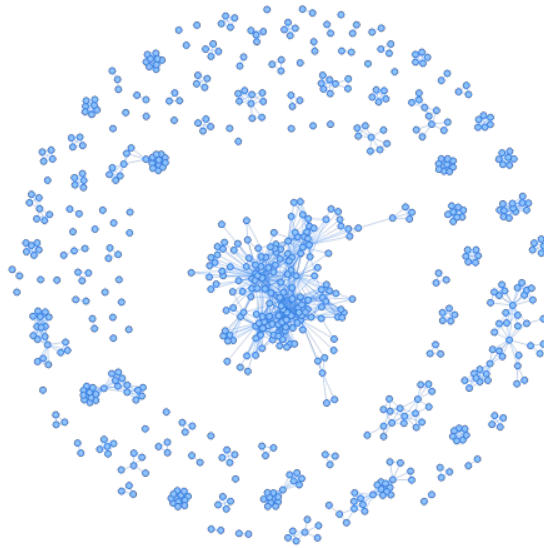


Network graph based on fruchtermanreingold layout

3.2 Preview of network graph with visNetwork

```
vis_data <- toVisNetworkData(network_graph)
vis_network <- visNetwork(nodes = vis_data$nodes, edges = vis_data$edges)

vis_network %>%
  visEdges(arrows = list(to = list(enabled= FALSE)), length=30) %>%
  visIgraphLayout(layout = "layout_nicely")
```



4. Answering Hypothesis

- **H1:** We expect more generalists to adopt the broker role in a network than specialists.

```
# Create data objects that are required for calculations
```

```
df_ipc_qc <- df_qc %>%
  mutate(IPC=strsplit(IPC, "\\n")) %>%
  unnest(IPC)
```

```
df_ipc_qc$IPC <- substring(df_ipc_qc$IPC, 1, 4)
```

```
unique_inventors <- unique(df_ipc_qc$Inventors)
```

```

df_h1 <- data.frame(Inventor=unique_inventors, Betw_Centr = 0, Degree_Centr=0,
  Degree_Centr_ZScore = 0, Herf_Index = 0, Herf_Index_ZScore = 0, IsBroker=
  "NotBroker", Diversity_Type="None", Applicant_Type="", IsPeripheralPlayer=
  "NotPeripheralPlayer")

# Measure betweenness centrality for each inventor to determine the brokerage
library(omnivor)

##
## Attache Paket: 'omnivor'

## Die folgenden Objekte sind maskiert von 'package:igraph':
##
##      degree_distribution, diameter

betw_centr <- betweenness_centrality(network_graph, normalized=T) # normalized!
df_betw_centr <- data.frame(as.list(betw_centr))
t_df_betw_centr <- transpose(df_betw_centr)
t_df_betw_centr$Inventor <- colnames(df_betw_centr)

for(inventor in unique_inventors){
  sub_df_qc <- df_qc[df_qc$Inventors == inventor, ]
  # Exclude inventors that have <= 1 patents published as there can be no
  # betweenness centrality measured
  if(nrow(sub_df_qc) > 1){
    centr <- t_df_betw_centr[t_df_betw_centr$Inventor == inventor, "V1"]
    df_h1[df_h1$Inventor == inventor, "Betw_Centr"] <- centr
  }
}

# Mark who is a broker
for(i in 1:nrow(df_h1)){
  row <- df_h1[i,]
  if((row$Betw_Centr > 0)){
    df_h1[df_h1$Inventor == row$Inventor, "IsBroker"] <- "Broker"
  }
}

# Calculate the Herf Index for each inventor
for(inventor in unique_inventors){
  df_ipc_qc_sub <- df_ipc_qc[df_ipc_qc$Inventors == inventor, ]
  ipc_table <- table(df_ipc_qc_sub$IPC)
  total_ipcs <- sum(ipc_table)
  table_as_df <- as.data.frame(ipc_table)
  herf_index <- 0
  for (i in 1:nrow(table_as_df)){
    row <- table_as_df[i,]
    val <- row$Freq
    herf_index <- herf_index + (val / total_ipcs)^2
  }
  df_h1[df_h1$Inventor == inventor, "Herf_Index"] <- herf_index
}

## Plot Herf Index distribution

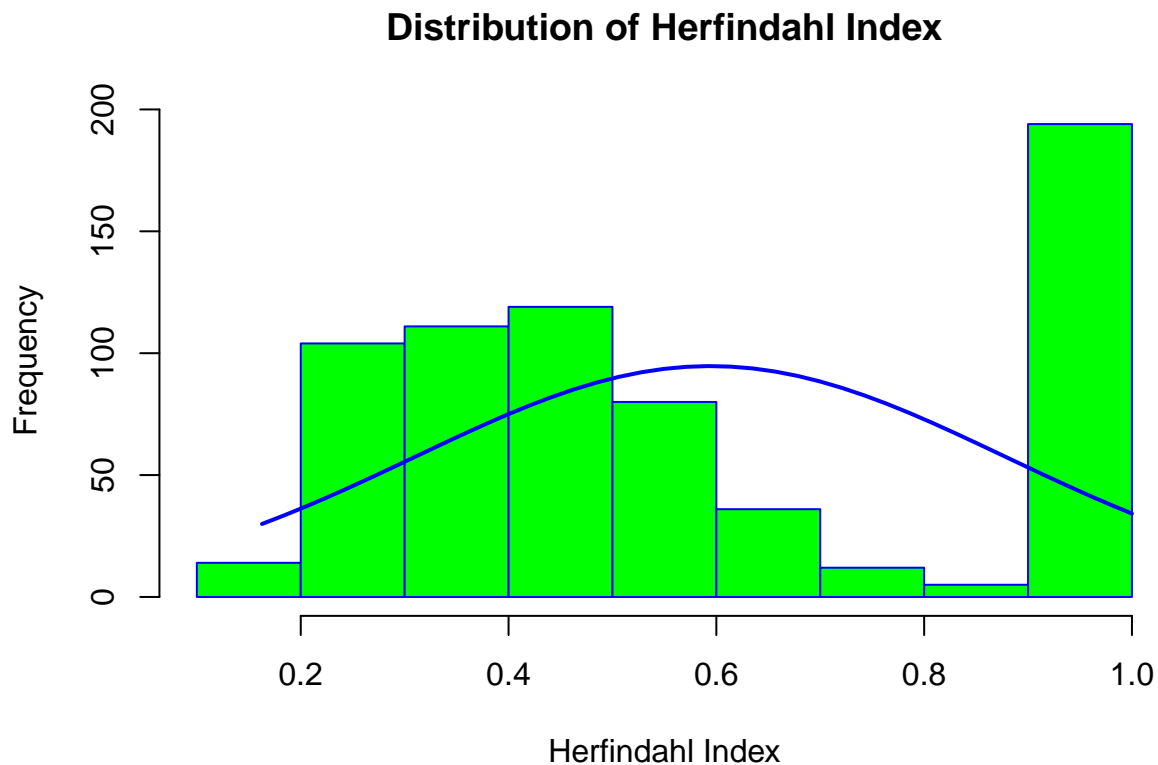
```

```

# Histogram overlaid with kernel density curve
x <- df_h1$Herf_Index
h <- hist(x, main="Distribution of Herfindahl Index", xlab="Herfindahl Index",
  border="blue", col="green", freq=T, ylab = "Frequency")

xfit <- seq(min(x), max(x), length=40)
yfit <- dnorm(xfit, mean=mean(x), sd=sd(x))
yfit <- yfit * diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)

```



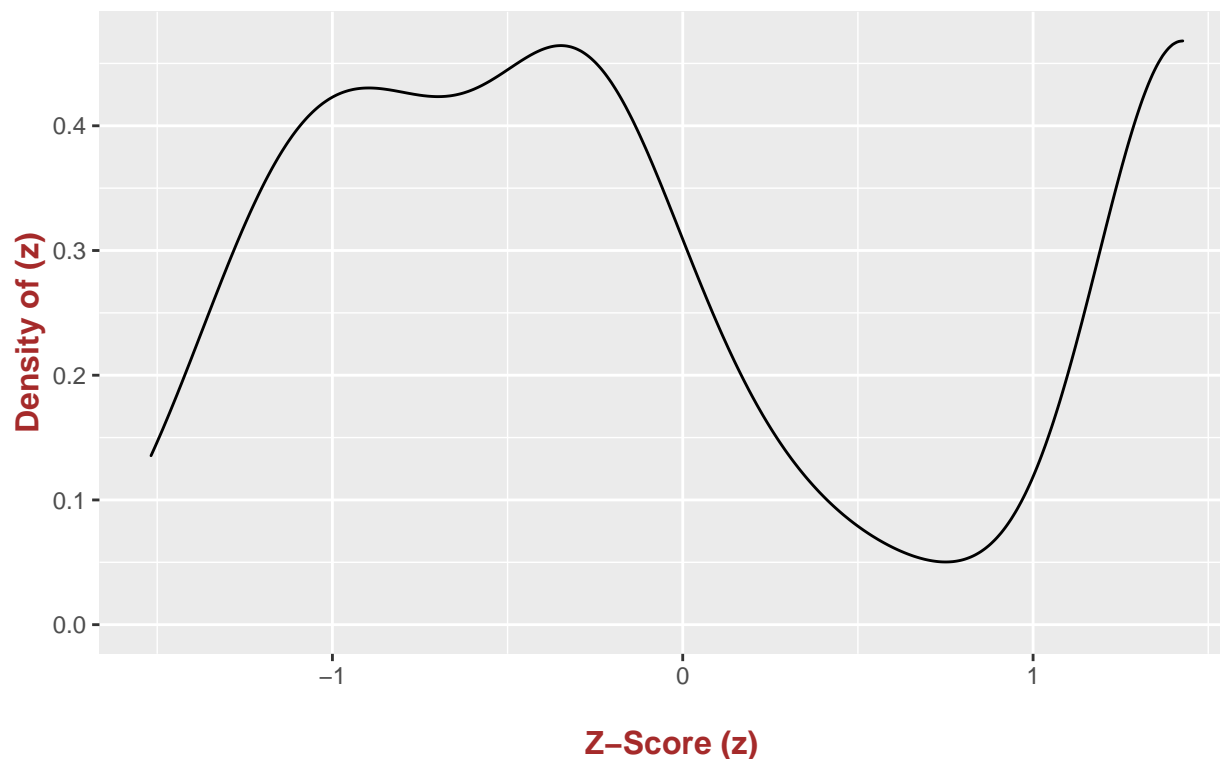
```

# Pre calculations before identifying generalists or specialists
# Calculate Z scores
# z score tells you how far a value is from the average of the data in terms
  of standard deviations.
df_h1 <- df_h1 %>% mutate(Herf_Index_ZScore = (Herf_Index - mean(Herf_Index))/
  sd(Herf_Index))

# Plot Distribution of Herf-Index Z-Scores
ggplot(df_h1, aes(x = Herf_Index_ZScore)) +
  geom_density() +
  labs(x = "\nZ-Score(z)", y = "Density of(z)", title = "Distribution of
  Herf-Index Z-Scores\n") +
  theme(plot.title = element_text(hjust = 0.5),
    axis.title.x = element_text(face="bold", colour="brown", size = 12),
    axis.title.y = element_text(face="bold", colour="brown", size = 12))

```

Distribution of Herf-Index Z-Scores



```
# Evaluate who is a Specialist
# H >= (Top 10% Herfindhal index z-score distribution)
# Calculate top 10%
#  $P(Z > c) = 0.1 \implies P(Z < c) = 0.9$ 
# use linear interpolation to approximate nearest values
# For further information, see: https://socratic.org/questions/what-is-the-z-score-corresponding-to-the-top-10-percent-of-any-data-set
nearest_value_z_top10_roof <- 1.29 # Z score for 0.9015, Standard normal table
nearest_value_z_top10_floor <- 1.28 # Z score 0.8997, Standard normal table
nearest_value_p_top10_roof <- 0.9015
nearest_value_p_top10_floor <- 0.8997

top10_way_0.9 <- ((0.9 - nearest_value_p_top10_roof) / (nearest_value_p_top10_
  floor - nearest_value_p_top10_roof))

z_score_top_10 <- nearest_value_z_top10_roof + ((nearest_value_z_top10_floor -
  nearest_value_z_top10_roof) * (top10_way_0.9))

raw_score_top_10 <- mean(df_h1$Herf_Index) + ((z_score_top_10) * sd(df_h1$Herf_
  _Index))

cat("Author with a Herf Index over ", raw_score_top_10, "are evaluated as part
  of the top 10%.")

## Author with a Herf Index over 0.9585829 are evaluated as part of the top
  10%.
```



```

for(i in 1:nrow(df_h1)){
  row <- df_h1[i,]
  sub_df_qc <- df_qc[df_qc$Inventors == row$Inventor, ]
  # Include the check after the amount of published patents for each inventor,
  # to avoid including inventors that have only published one patent, where
  # no clear IPC classes diversity is observable
  if((row$Herf_Index >= raw_score_top_10) & (nrow(sub_df_qc) > 1)){
    df_h1[df_h1$Inventor == row$Inventor, "Diversity_Type"] <- "Specialist"
  }
}

# Evaluate who is a Generalist
# Lowest 10% Herfindhal index distribution
# For more information, see: https://www.dummies.com/article/academics-the-arts/math/statistics/how-to-find-a-percentile-for-a-normal-distribution-169600/
# Use linear interpolation to capture edge cases
nearest_value_z_lowest10_roof <- -1.28 # Z score for 0.1003, Standard normal table
nearest_value_z_lowest10_floor <- -1.29 # Z score for 0.0985, Standard normal table
nearest_value_p_lowest10_roof <- 0.1003
nearest_value_p_lowest10_floor <- 0.0985

lowest10_way_0.1 <- ((0.1 - nearest_value_p_lowest10_roof) / (nearest_value_p_lowest10_floor - nearest_value_p_lowest10_roof))

z_score_lowest_10 <- nearest_value_z_lowest10_roof + ((nearest_value_z_lowest10_floor - nearest_value_z_lowest10_roof) * (lowest10_way_0.1))

raw_score_lowest_10 <- mean(df_h1$Herf_Index) + ((z_score_lowest_10) * sd(df_h1$Herf_Index))

cat("Author with a Herf Index under", raw_score_lowest_10, "are evaluated as part of the lowest 10%.")

## Author with a Herf Index under 0.2298311 are evaluated as part of the lowest 10%.

for(i in 1:nrow(df_h1)){
  row <- df_h1[i,]
  sub_df_qc <- df_qc[df_qc$Inventors == row$Inventor, ]
  # Include the check after the amount of published patents for each inventor,
  # to avoid including inventors that have only published one patent, where
  # no clear IPC classes diversity is observable
  if((row$Herf_Index <= raw_score_lowest_10) & (nrow(sub_df_qc) > 1)){
    df_h1[df_h1$Inventor == row$Inventor, "Diversity_Type"] <- "Generalist"
  }
}

# Connect brokerage and generalist/specialist distribution
broker_diversity <- df_h1[df_h1$IsBroker == "Broker" & df_h1$Diversity_Type != "None",]

```

```

cat("In total", nrow(broker_diversity) ,"brokers with given diversity types
    were identified.")

## In total 30 brokers with given diversity types were identified.

print(table(broker_diversity$Diversity_Type))

##
## Generalist Specialist
##          20          10

cat("From these brokers, we identified 20 Generalists, which equals to a
    distribution of", (20/nrow(broker_diversity)) * 100, "%\n")

## From these brokers, we identified 20 Generalists, which equals to a
    distribution of 66.66667 %

cat("From these brokers, we identified 10 Specialists, which equals to a
    distribution of", (10/nrow(broker_diversity)) * 100, "%")

## From these brokers, we identified 10 Specialists, which equals to a
    distribution of 33.33333 %

print("There are more generalists than specialists based on the broker
    distribution.")

## [1] "There are more generalists than specialists based on the broker
    distribution."

# Visualize key players
nodes_groups <- vector()

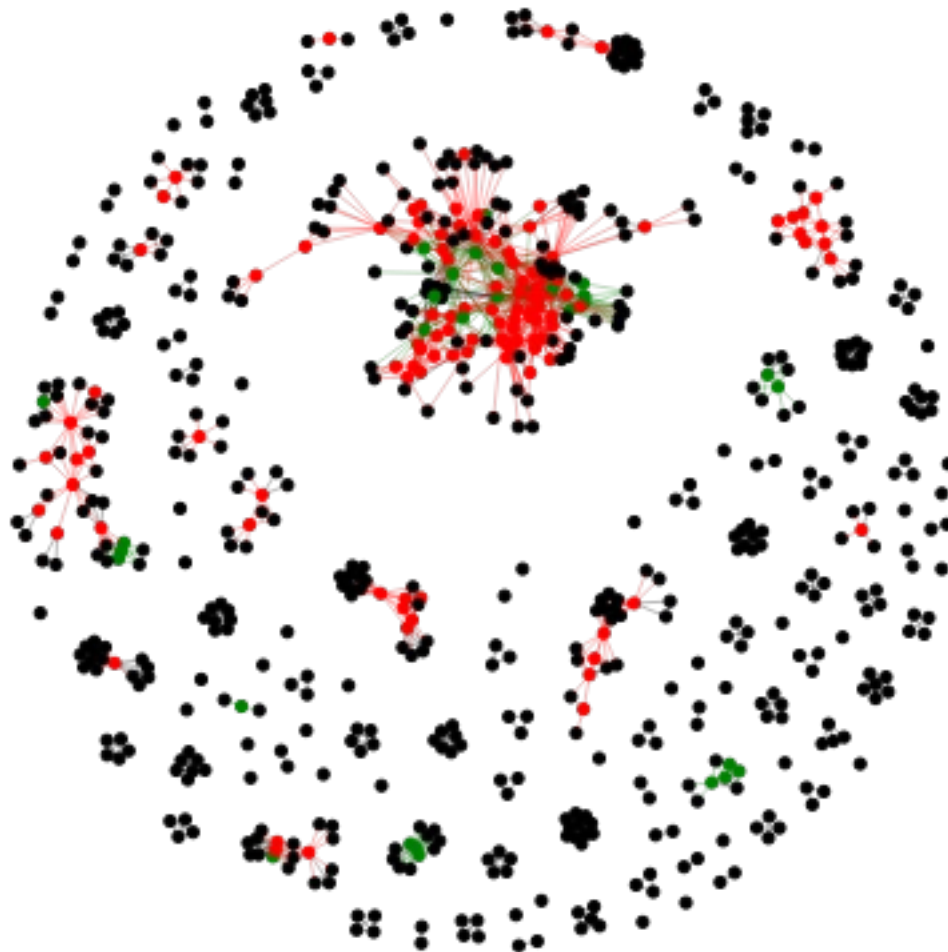
for (current_node in vis_data$nodes$id){
  if (current_node %in% df_h1$Inventor[df_h1$IsBroker == "Broker"]){
    if(current_node %in% broker_diversity$Inventor){
      nodes_groups <- c(nodes_groups, "Green")
    }else{
      nodes_groups <- c(nodes_groups, "Red")
    }
  }else{
    nodes_groups <- c(nodes_groups, "Black")
  }
}

vis_data$nodes$color <- nodes_groups

vis_network_2 <- visNetwork(nodes = vis_data$nodes, edges = vis_data$edges,
    width = "100%")

# Red Nodes = Broker in general
# Green Nodes = Broker with Diversity Type
# Black Nodes = Others
vis_network_2 %>%
  visPhysics(solver = "forceAtlas2Based", forceAtlas2Based = list(
    gravitationalConstant = -100, avoidOverlap = 1)) %>%
  visIgraphLayout(layout = "layout_nicely", physics = FALSE, smooth=F)

```



-
- H2: We expect generalist inventors of academic origin to adopt the broker role more often than generalist inventors of industrial or private origin.

```

# We are continuing using the dataframe from H1.
# Determine origin of each inventor / view & use dataframe "edge_list"
# Selecting of applicant type is based on the majority of type occurrences,
  see function "identify_applicant_type" in 2.1
df_h2 <- data.frame(from=edge_list$from, to=edge_list$to, group=edge_list$
  group)

for(i in 1:nrow(df_h1)){
  row <- df_h1[i,]
  df_sub <- df_h2[df_h2$from == row$Inventor, ]
  df_h1[df_h1$Inventor == row$Inventor, "Applicant_Type"] <- names(which.max(
    table(df_sub$group)))
}

table_applicant_type <- table(df_h1$Applicant_Type)
cat("Overall Distribution of applicant types through all Inventors")

## Overall Distribution of applicant types through all Inventors

print(table_applicant_type)

##
## academic industrial private
## 65 514 96

df_applicant_type <- as.data.frame(table_applicant_type)
df_applicant_type$Distribution <- df_applicant_type$Freq / sum(df_applicant_
  type$Freq)

# Calculate distribution of diversities types of the identified generalist who
  are brokers
generalist_with_applicant_type <- df_h1[df_h1$IsBroker == "Broker" & df_h1$
  Diversity_Type == "Generalist",]

# Output results in numbers
print(table(generalist_with_applicant_type$Applicant_Type))

##
## academic industrial
## 3 17

amount_generalists <- nrow(generalist_with_applicant_type)

cat("From", amount_generalists, "generalists, we identified 3 of academic
  origin, which equals to a distribution of", (3/amount_generalists) * 100,
  "%\n")

## From 20 generalists, we identified 3 of academic origin, which equals to a
  distribution of 15 %

cat("From", amount_generalists, "generalists, we identified 17 of industrial
  origin, which equals to a distribution of", (17/amount_generalists) * 100,
  "%\n")

```

```

## From 20 generalists , we identified 17 of industrial origin , which equals to
  a distribution of 85 %

# Draw Conclusion from subset on larger distribution by utilizing the
  correlation coefficient based on pearson 's method
pairwise_comp <- df_h1[c(7,8,9)]

pairwise_comp$IsBroker <- factor(pairwise_comp$IsBroker)
pairwise_comp$Diversity_Type <- factor(pairwise_comp$Diversity_Type)
pairwise_comp$Applicant_Type <- factor(pairwise_comp$Applicant_Type)

pairwise_comp <- pairwise_comp[pairwise_comp$IsBroker == "Broker" ,]

# Convert to bivariate binary distribution (Diversity and Applicant Type), as
  the data set is based on only brokers
mm <- model.matrix(~ IsBroker + Diversity_Type + Applicant_Type, data=pairwise
  _comp, contrasts.arg = lapply(pairwise_comp[, sapply(pairwise_comp, is.
    factor)], drop = FALSE), contrasts, contrasts = FALSE))

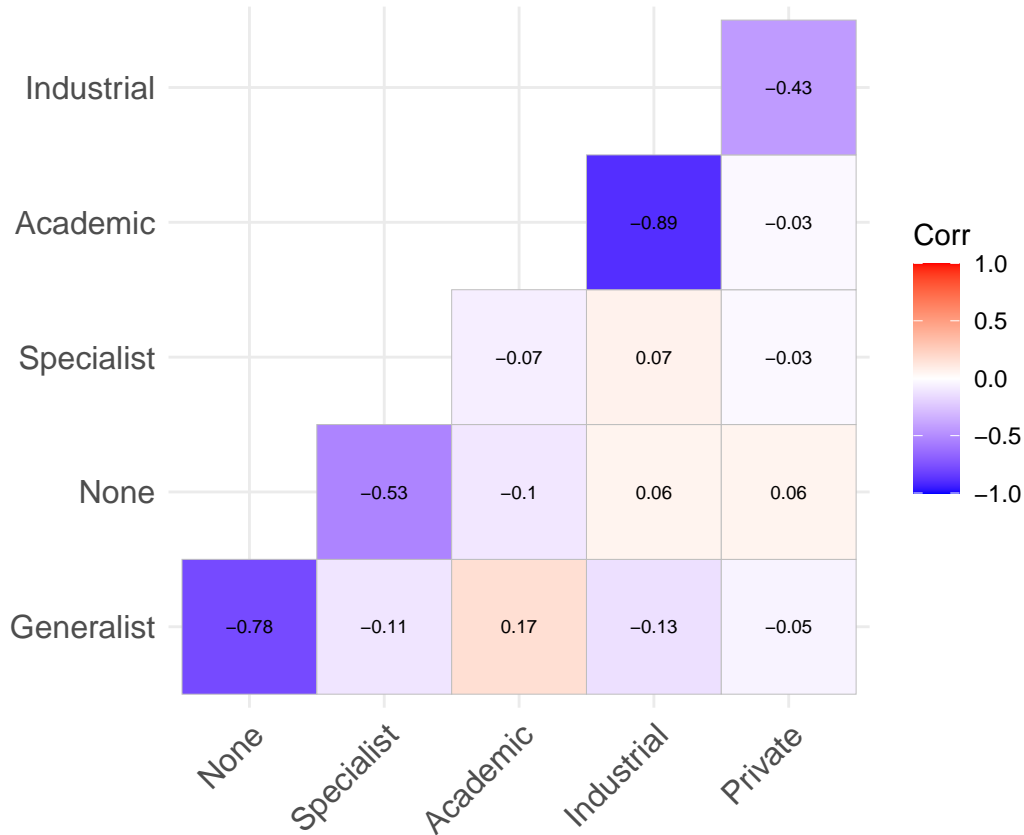
# pairwise.complete.obs means that the correlation or covariance between each
  pair of variables is computed using all complete pairs of observations on
  those variables
mm_cor <- cor(mm, use="pairwise.complete.obs", method = c("pearson"))

## Warning in cor(mm, use = "pairwise.complete.obs", method = c("pearson")):
## Standardabweichung ist Null

colnames(mm_cor) <- c("(Intercept)", "Broker", "NotBroker", "Generalist", "
  None", "Specialist", "Academic", "Industrial", "Private")
rownames(mm_cor) <- c("(Intercept)", "Broker", "NotBroker", "Generalist", "
  None", "Specialist", "Academic", "Industrial", "Private")

mm_cor %>% ggcorrplot(show.diag = F, type = "lower", lab=TRUE, lab_size=2.5)

```



```
cat("Correlation of Generalist and Academic is 0.17, given that the inspected  
inventors are brokers.")
```

```
## Correlation of Generalist and Academic is 0.17, given that the inspected  
inventors are brokers.
```

```
cat("Correlation of Generalist and Industrial is -0.13, given that the  
inspected inventors are brokers.")
```

```
## Correlation of Generalist and Industrial is -0.13, given that the inspected  
inventors are brokers.
```

```
cat("Despite the measurement of", amount_generalists, "generalists, where 3  
inventors are of academic origin, the overall correlation factor of  
generalist in combination with academic is higher.")
```

```
## Despite the measurement of 20 generalists, where 3 inventors are of  
academic origin, the overall correlation factor of generalist in  
combination with academic is higher.
```

-
- **H3: We expect peripheral players to be specialists.**

```

# Calculate out-degree centrality for each inventor
degree_centrality <- degree(network_graph, mode = "out", normalized = T)
df_degree_centr <- data.frame(as.list(degree_centrality))
t_df_degree_centr <- transpose(df_degree_centr)
t_df_degree_centr$Inventor <- colnames(df_degree_centr)

# Add out-degree centrality to df_h1
for(i in 1:nrow(t_df_degree_centr)){
  row <- t_df_degree_centr[i,]
  df_h1[df_h1$Inventor == row$Inventor, "Degree_Centr"] <- row$V1
}

# Calculate Z scores for Degree_Centr
df_h1 <- df_h1 %>% mutate(Degree_Centr_ZScore = (Degree_Centr - mean(Degree_
  Centr))/sd(Degree_Centr))

# Adapt approach from H1 towards calculating lowest 10%
# Lowest 30% of the degree centrality, as distribution indicates that there
  are few inventors with way more connections than the majority.
# This is indicated by the fact, that the standard deviation is larger than
  the mean.
# Use linear interpolation to capture edge cases
nearest_value_z_lowest30_roof <- -0.52 # Z score for 0.30153, Standard normal
  table
nearest_value_z_lowest30_floor <- -0.53 # Z score for 0.29806, Standard normal
  table
nearest_value_p_lowest30_roof <- 0.30153
nearest_value_p_lowest30_floor <- 0.29806

lowest30_way_0.3 <- ((0.3 - nearest_value_p_lowest30_roof) / (nearest_value_p_
  lowest30_floor - nearest_value_p_lowest30_roof))

z_score_lowest_30 <- nearest_value_z_lowest30_roof + ((nearest_value_z_
  lowest30_floor - nearest_value_z_lowest30_roof) * (lowest30_way_0.3))

mean_degree_centr <- (mean(df_h1$Degree_Centr))
cat("Mean of degree centrality distribution:", mean_degree_centr, "\n")

## Mean of degree centrality distribution: 0.008567975

sd_degree_centr <- sd(df_h1$Degree_Centr)
cat("Standard Deviation of degree centrality distribution:", sd_degree_centr, "\
  n")

## Standard Deviation of degree centrality distribution: 0.008695654

print("Standard deviation is larger than mean.")

## [1] "Standard deviation is larger than mean."

raw_score_lowest_30_degree_centr <- mean_degree_centr + (z_score_lowest_30 *
  sd_degree_centr)

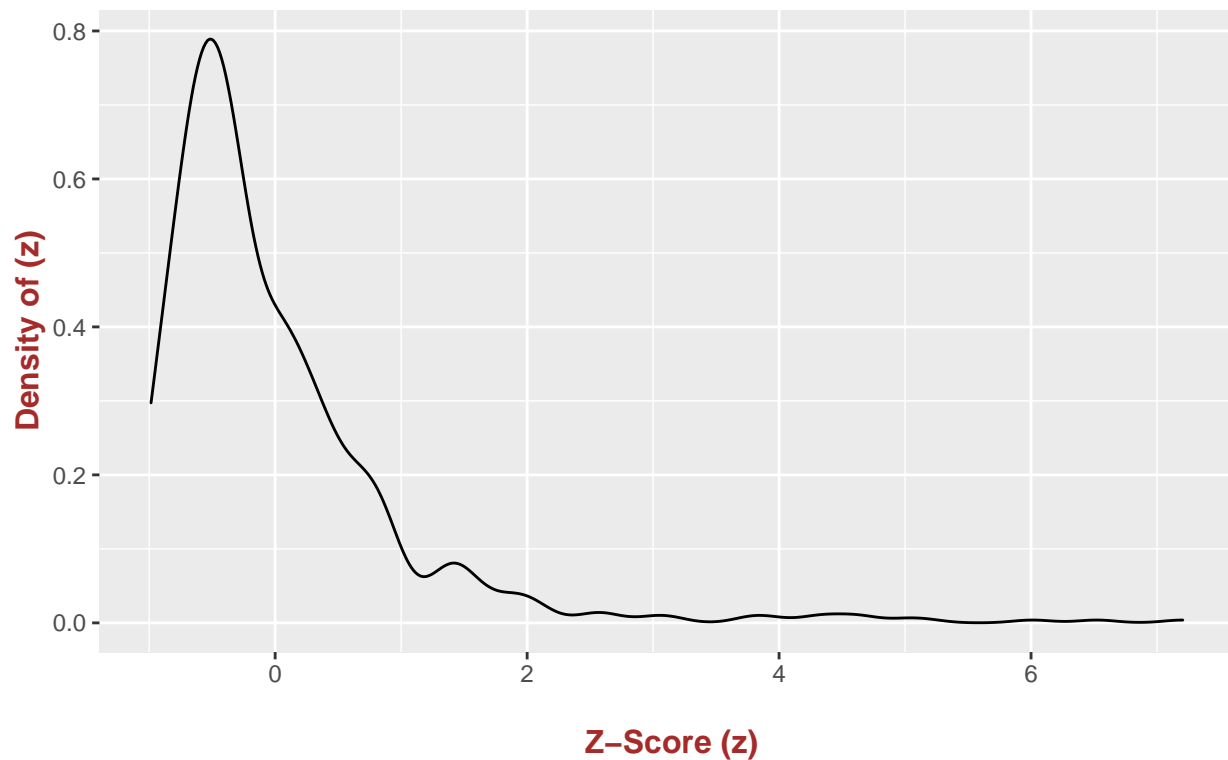
cat("Author with a Degree Centrality of under", raw_score_lowest_30_degree_
  centr, "are evaluated as part of the lowest 30% and are therefore
  peripheral players.")

```

```
## Author with a Degree Centrality of under 0.004007893 are evaluated as part
of the lowest 30% and are therefore peripheral players.
```

```
# Plot Distribution of Degree Centrality Z-Scores
ggplot(df_h1, aes(x = Degree_Centr_ZScore)) +
  geom_density() +
  labs(x = "\nZ-Score(z)", y = "Density of(z)", title = "Distribution of
Degree Centrality Z-Scores\n") +
  theme(plot.title = element_text(hjust = 0.5),
        axis.title.x = element_text(face="bold", colour="brown", size = 12),
        axis.title.y = element_text(face="bold", colour="brown", size = 12))
```

Distribution of Degree Centrality Z-Scores



```
# Get the distribution of generalist/specialist based on the peripheral
players
for(i in 1:nrow(df_h1)){
  row <- df_h1[i,]
  sub_df_qc <- df_qc[df_qc$Inventors == row$Inventor, ]
  if(row$Degree_Centr <= row_score_lowest_30_degree_centr){
    df_h1[df_h1$Inventor == row$Inventor, "IsPeripheralPlayer"] <- "
PeripheralPlayer"
  }
}

peripheral_diversity <- df_h1[df_h1$IsPeripheralPlaye == "PeripheralPlayer" &
df_h1$Diversity_Type != "None",]

print(table(peripheral_diversity$Diversity_Type))
```



```

##
## Generalist Specialist
##           2           5

amount_relevant_peripheral_p <- nrow(peripheral_diversity)

cat("From", amount_relevant_peripheral_p, "peripheral_players, we identified 2
    generalists, which equals to a distribution of", (2/amount_relevant_
    peripheral_p) * 100, "%\n")

## From 7 peripheral players, we identified 2 generalists, which equals to a
    distribution of 28.57143 %

cat("From", amount_relevant_peripheral_p, "peripheral_players, we identified 5
    specialists, which equals to a distribution of", (5/amount_relevant_
    peripheral_p) * 100, "%\n")

## From 7 peripheral players, we identified 5 specialists, which equals to a
    distribution of 71.42857 %

print("Based on this measurement and the awareness of the small sample size, a
    confirmation for our hypothesis can be derived.")

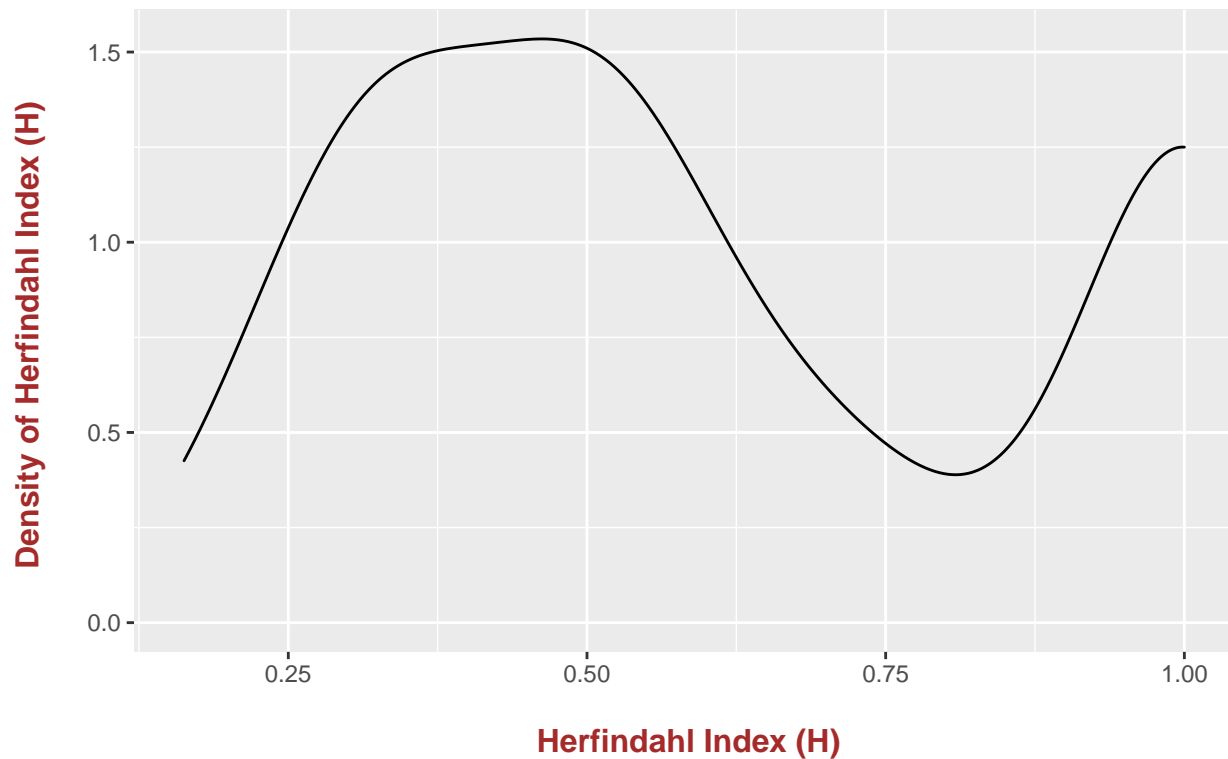
## [1] "Based on this measurement and the awareness of the small sample size,
    a confirmation for our hypothesis can be derived."

# Get the distribution of Herf Index based on the peripheral players
df_peripheral <- df_h1[df_h1$IsPeripheralPlaye == "PeripheralPlayer",]

ggplot(df_peripheral, aes(x = Herf_Index)) +
  geom_density() +
  labs(x = "\nHerfindahl_Index(H)",
    y = "Density of Herfindahl_Index(H)\n",
    title = "Distribution of Herf_Index based on peripheral_players\n") +
  theme(plot.title = element_text(hjust = 0.5),
    axis.title.x = element_text(face="bold", colour="brown", size = 12),
    axis.title.y = element_text(face="bold", colour="brown", size = 12))

```

Distribution of Herf Index based on peripheral players



```
print("Interpretation of Herfindahl Index Distribution based on existing
peripheral players:")

## [1] "Interpretation of Herfindahl Index Distribution based on existing
peripheral players:"

print("There is a trends towards a lower herf index value observable.")

## [1] "There is a trends towards a lower herf index value observable."

print("This allows us to derive an orientation of peripheral players towards
the generalists segment. This is contrary to our stated hypothesis.")

## [1] "This allows us to derive an orientation of peripheral players towards
the generalists segment. This is contrary to our stated hypothesis."

# Visualize results
vis_data_h3 <- toVisNetworkData(network_graph)
vis_network_h3 <- visNetwork(nodes = vis_data_h3$nodes, edges = vis_data_h3$
edges)

nodes_groups_h3 <- vector()

for (current_node in vis_data_h3$nodes$id){
  if (current_node %in% df_h1$Inventor[df_h1$IsPeripheralPlayer == "
PeripheralPlayer"]){
    nodes_groups_h3 <- c(nodes_groups_h3, "Blue")
  }
}
```

```

    }else{
      nodes_groups_h3 <- c(nodes_groups_h3, "Black")
    }
  }

vis_data_h3$nodes$color <- nodes_groups_h3

vis_network_h3 <- visNetwork(nodes = vis_data_h3$nodes, edges = vis_data_h3$
  edges, width = "100%")

# Blue Nodes = Peripheral Players
# Black Nodes = Not Peripheral Players
vis_network_h3 %>%
  visPhysics(solver = "forceAtlas2Based",
    forceAtlas2Based = list(gravitationalConstant = -100,
      avoidOverlap = 1)) %>%
  visIgraphLayout(layout = "layout_nicely", physics = FALSE, smooth=F)

```

