Final Project

Jerin Jacob

02/22/2023

Introduction

We are living in a world that produces a huge volume of waste everyday. It is estimated that by 2050, the global waste produced will be more than 3.4 billion tons every year. Certain industries produce large volume of waste while some other industries are considered to be cleaner than others. The world has already moved towards recycling as a part of reducing waste dumped in overall. Waste materials produced by certain industries can be used as raw material for certain other industries. This project is an attempt to study the input-output data of materials between industries and the categories of wastes each industries produce. The dataset is from the 'Waste Input Output Analysis' by Nakamura, S. and Kondo, Yasushi. It is a data from Japan and therefore the economic flow is given in 1 million Japanese yen. The analysis will help us to find which all industries serve how many other industries with the goods they produce and compare it with the waste emission by each of those industires.

Research Question

How the most influencial industries in terms of their interaction to other industries contribute to the wastes produced?

Reading the Data

```
data_2011 <- read_xlsx("_data/Project_data/WIO_2011.xlsx", sheet = "WIOdata")
```

```
## New names:
## * `` -> `...1`
```

```
head(data_2011)
#data_2011
dim(data_2011)
```

```
## [1] 294 103
```

The dataset has 294 rows and 103 columns. We are interested in only the output flow between industries and the waste flow from industries to different waste management processes. Therefore, we can trim the data as a subset which is in the form we want.
Cleaning Data

```
df <- data_2011[1:81, 1:92]
head(df)
industry_io <- data_2011[1:81, 1:82]
waste_io <- data_2011[1:81, c(1, 83:92)]
head(waste_io)
```

Creating Network

After cleaning the dataset, next step is to create network data out of it.

```
#df <- industry_io
# Transform the data to long format
df_long <- melt(df, id.vars = "...1", variable.name = "to", value.name = "weight", variable.factor = FA

# Rename the "Industries" column to "From"
colnames(df_long)[colnames(df_long) == "...1"] <- "from"

# Drop rows with weight 0
df_long <- df_long[df_long$weight != 0, ]

df_long <- df_long |>
  mutate(waste_process = ifelse(to %in% c("Incineration", "Dehydration", "Concentration", "Shredding",
                                  "Feed conversion", "Gasification", "Refuse derived fuel", "Landfill")
head(df_long)
dim(df_long)
```

```
## [1] 4167    4
```

There are negative values in the 'weight' column. When the value is negative in directed network, it could be probably because the transaction was done in the reverse direction. So, assuming likewise, we can swap the from and to where weight is negative and then get the absolute values for weight so that we don't want to deal with anymore negative values!

```
str(df_long)
```

```
## 'data.frame':    4167 obs. of  4 variables:
##  $ from         : chr  "Crop cultivation" "Livestock" "Agricultural services" "Forestry" ...
##  $ to           : Factor w/ 91 levels "Crop cultivation",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ weight       : num  183568 47771 337249 1321 46056 ...
##  $ waste_process: num  0 0 0 0 0 0 0 0 0 0 ...
```

```
df_long$to <- as.character(df_long$to)


df_long <- df_long %>%
  mutate(new_from = ifelse(weight < 0, to, from),
         new_to = ifelse(weight < 0, from, to)) %>%
  select(-c(from, to)) %>%
  mutate(weight = abs(weight)) %>%
  rename(from = new_from, to = new_to)
df_long <- df_long[, c("from", "to", "weight", "waste_process")]


df_long |>
  filter(waste_process == 1) |>
  head()
```

Creating the network

```
g_df <- graph_from_data_frame(df_long)
# Extract the weighted vertex attribute values from the dataframe
#vertex_attributes <- df[, 82:92]
```

```
E(g_df)$weight <- df_long$weight
#E(g_df)$waste_process <- df_long$waste_process

process_names <- c("Incineration", "Dehydration", "Concentration", "Shredding", "Filtration", "Compostin
                   "Feed conversion", "Gasification", "Refuse derived fuel", "Landfill")

# Create an empty vector to store the attribute values
vertex_attribute <- rep("industry", vcount(g_df))

# Find the vertices with names in the list and assign attribute value of "waste processing"
matching_vertices <- which(V(g_df)$name %in% process_names)
vertex_attribute[matching_vertices] <- "waste processing"

# Add the vertex attribute to the graph
V(g_df)$process <- vertex_attribute


#V(g_df)$process

ls(df_long)
```

```
## [1] "from"          "to"            "waste_process" "weight"
```

```
#plot(g_df)
```

Describing the Network Data

```
vcount(g_df)
```

```
## [1] 90
```

```
ecount(g_df)
```

```
## [1] 4167
```

```
is_bipartite(g_df)
```

```
## [1] FALSE
```

```
is_directed(g_df)
```

```
## [1] TRUE
```

```
is_weighted(g_df)
```

```
## [1] TRUE
```

The network has 90 vertices and 4167 edges.

```
summary(E(g_df)$weight)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##        0      467     5410   110218    36673 11899111
```

```
vertex_attr_names(g_df)
```

```
## [1] "name"    "process"
```

```r
edge_attr_names(g_df)
```

```
## [1] "weight"        "waste_process"
```

Dyad & Triad Census

```r
igraph::dyad.census(g_df)
```

```
## $mut
## [1] 1064
##
## $asym
## [1] 1951
##
## $null
## [1] 990
```

```r
igraph::triad_census(g_df)
```

```
##  [1]  7680 10472  2204 21177  1242  2515   663 11570 11335   188  1561  2147 21579  2396 12177  8574
```

```r
sum(igraph::triad_census(g_df))
```

```
## [1] 117480
```

```r
(90*89*88)/(3*2)
```

```
## [1] 117480
```

Transitivity/ Global Clustering

Let us look at the clustering pattern in a global level of network

```r
transitivity(g_df)
```

```
## [1] 0.818957
```

The network has a high level of transitivity. It means, when two nodes are connected to a neighbouring node, it is highly likely that all of them are connected to each other.

Local Transitivity / Clustering

Now let us look at the Local transitivity. Since the number of vertices are fairly high, we will look on the average clustering coefficient rather than local clustering coefficient.

```r
transitivity(g_df, type = "average")
```

```
## [1] 0.8658444
```

The average clustering coefficient of 0.8658 suggests that the network is highly clustered. This means that the network has tightly connected sub groups/clusters.

Path Length and Geodesic

```r
average.path.length(g_df, directed = F)
```

```
## [1] 7.457572
```

The average path length of the network is 7.45.

Component Structure and Membership

```
names(igraph::components(g_df)) # Elements of components
```

```
## [1] "membership" "csize"      "no"
```

```
igraph::components(g_df)$no
```

```
## [1] 1
```

```
igraph::components(g_df)$csize
```

```
## [1] 90
```

```
#igraph::components(g_df)$membership
```

The component structure shows that there is only one big component with 90 members. In other words, there is no isolates in this network.

Density of Network

```
graph.density(g_df)
```

```
## [1] 0.5202247
```

The network has a 0.5202 density which means 0.5202 proportion of all possible ties are present in this network.

Creating a Dataframe with the Vertex Degree values

```
df_degree <- data.frame(name = V(g_df)$name, degree = igraph::degree(g_df))
head(df_degree)
```

In degree and out degree

```
df_degree <- df_degree |>
  mutate(indegree = igraph::degree(g_df, mode = "in"),

         outdegree = igraph::degree(g_df, mode = "out"))
df_degree |>
  arrange(desc(outdegree)) |>
  slice(1:5)

df_degree |>
  arrange(desc(indegree)) |>
  slice(1:5)
```

Summary statistics of Network Degree

```
summary(df_degree)
```

```
##      name               degree          indegree       outdegree
##  Length:90          Min.   : 38.00   Min.   :17.00   Min.   : 0.00
##  Class :character   1st Qu.: 57.75   1st Qu.:39.25   1st Qu.:10.25
```

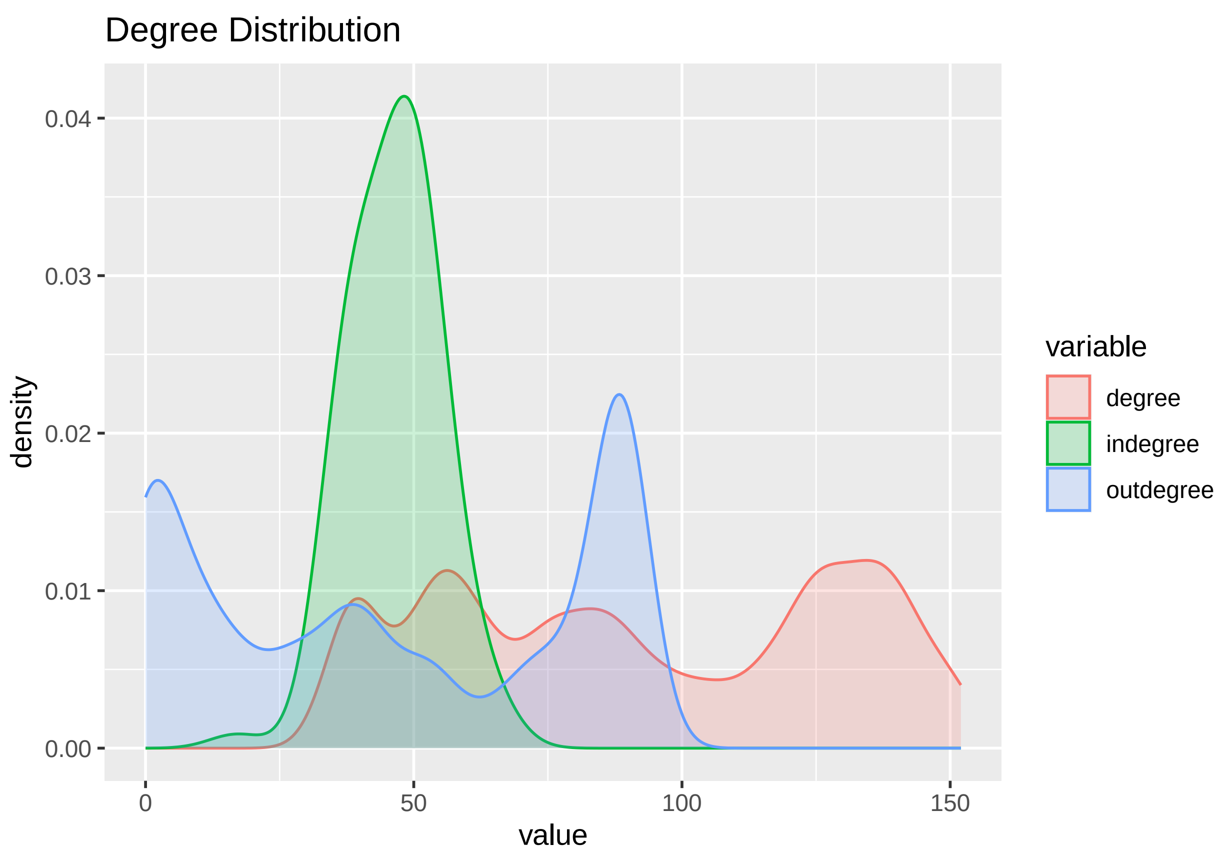
```
##   Mode   :character   Median : 89.00   Median :48.00   Median :41.00
##                       Mean   : 92.60   Mean   :46.30   Mean   :46.30
##                       3rd Qu.:126.75   3rd Qu.:52.00   3rd Qu.:86.00
##                       Max.   :152.00   Max.   :66.00   Max.   :92.00
```

Network Degree Distribution

```
df_degree %>% melt %>% filter(variable != 'output' & variable != 'eigen.centrality') %>%
  ggplot(aes(x = value, fill = variable, color = variable)) + geom_density(alpha = .2, bw = 5) +
  ggtitle('Degree Distribution')
```

```
## Using name as id variables
```

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



The distribution of degrees shows that the indegree values are more at a level of 50.

Network Degree Centralization

```
centr_degree(g_df, mode = "in")$centralization
```

```
## [1] 0.2213483
```

```
centr_degree(g_df, mode = "out")$centralization
```

```
## [1] 0.5134831
```

Eigen Vector

```
temp_eigen <- centr_eigen(g_df, directed = T)
names(temp_eigen)
```

```
## [1] "vector"          "value"           "options"         "centralization"  "theoretical_max"
```

```
length(temp_eigen$vector)
```

```
## [1] 90
```

```
temp_eigen$vector
```

```
##  [1] 0.5953827 0.5120362 0.6280237 0.6805738 0.5954991 0.6824618 0.6057831 0.8187135 0.7071076 0.6528
## [11] 0.6034703 0.6478165 0.7163664 0.7881786 0.5418807 0.7574951 0.6661287 0.6401862 0.6576396 0.7781
## [21] 0.6086314 0.7930444 0.7667579 0.7641407 0.7718081 0.8899521 0.7727183 0.5146864 0.5145222 0.6882
## [31] 0.6419725 0.5519556 0.5175506 0.9087712 0.7717444 0.7442549 0.5454263 0.8646528 0.9102065 0.2697
## [41] 0.7511301 0.7086050 0.6025945 0.6784769 0.6992363 0.7370351 0.7233416 0.7887455 0.7168331 0.6934
## [51] 0.7050182 0.7535558 0.8082977 0.8390455 0.6841809 0.7379447 0.7483575 0.7978797 0.7380666 0.5369
## [61] 0.5485777 0.5782104 0.4740734 0.5273760 0.6535068 0.5117106 0.5468021 1.0000000 0.7119895 0.7187
## [71] 0.7131842 0.7267811 0.7680584 0.7145214 0.6885328 0.8205471 0.8157092 0.7580999 0.9623841 0.6121
## [81] 0.8306890 0.5704533 0.5704533 0.5594488 0.5594488 0.5594488 0.5594488 0.5594488 0.5594488 0.5704
```

```
temp_eigen$centralization
```

```
## [1] 0.3226412
```

Adding Eigen Vector to node level measures dataframe

```
df_degree$eigen <- centr_eigen(g_df, directed = T)$vector
#df_degree
arrange(df_degree, desc(eigen)) |> slice(1:5)
```

Bonacich Power Centrality to the dataframe

```
df_degree$bonpow <- power_centrality(g_df)

df_degree |>
  arrange(desc(bonpow)) |>
  slice(1:5)
```

Derived and Reflected Centrality

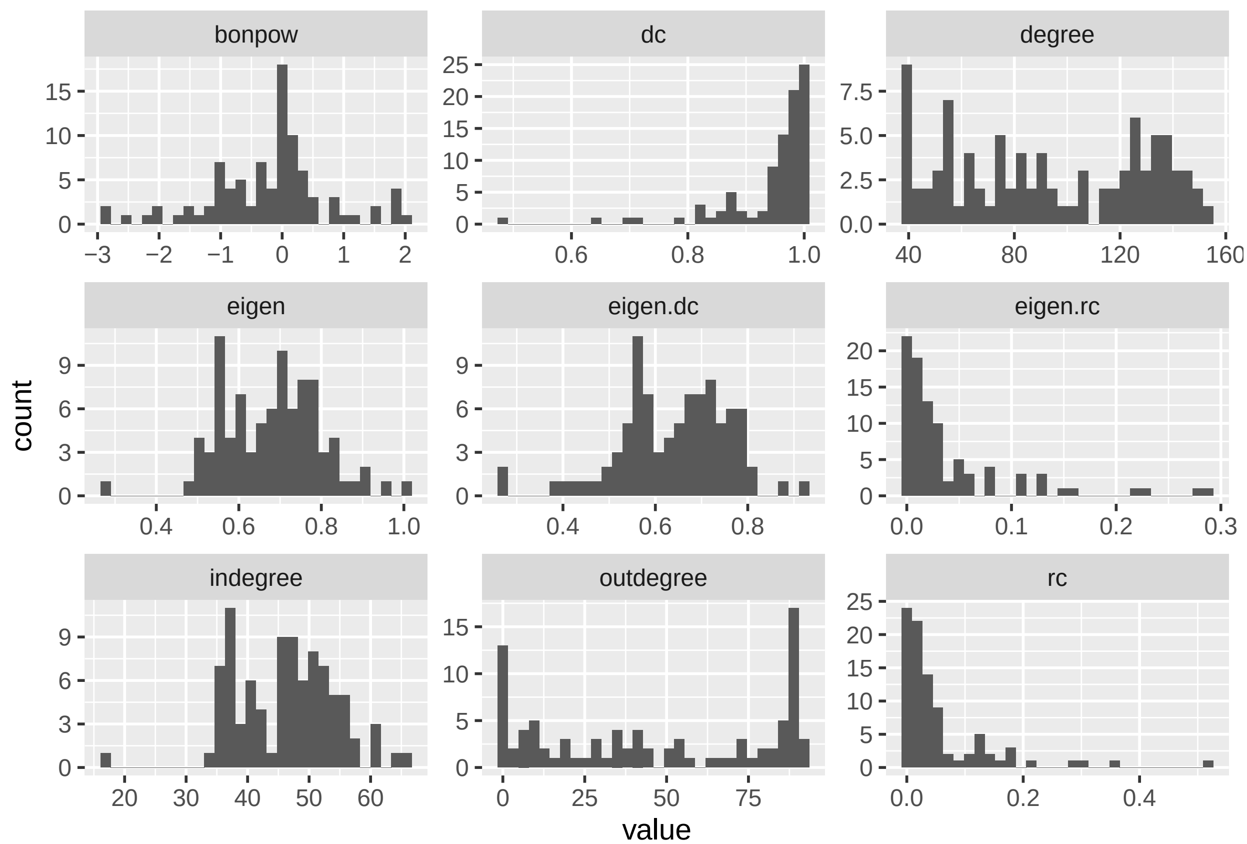
```
matrix_df_degree <- as.matrix(as_adjacency_matrix(g_df, attr = "weight"))
# Square the adjacency matrix
matrix_df_degree_sq <- t(matrix_df_degree) %*% matrix_df_degree
# Calculate the proportion of reflected centrality
df_degree$rc <- diag(matrix_df_degree_sq)/rowSums(matrix_df_degree_sq)

# Replace missing values with 0
df_degree$rc <- ifelse(is.nan(df_degree$rc), 0, df_degree$rc)

# Calculate received eigen value centrality

df_degree$eigen.rc <- df_degree$eigen * df_degree$rc
```

```r
# Calculate the proportion of derived centrality
df_degree$dc <- 1-diag(matrix_df_degree_sq)/rowSums(matrix_df_degree_sq)

# Replace missing values with 0
df_degree$dc <- ifelse(is.nan(df_degree$dc), 0, df_degree$dc)

# Calculate derived eigen value centrality

df_degree$eigen.dc <- df_degree$eigen * df_degree$dc
```

Centrality Score Distribution

```r
df_degree |>
  select(-name) |>
  gather() |>
  ggplot(aes(value)) +
  geom_histogram() +
  facet_wrap(~key, scales = "free")
```

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

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Centrality Measure Correlations

Closeness Centrality

```r
df_degree$close <- igraph::closeness(g_df)

df_degree |>
  arrange(desc(close)) |>
  slice(1:5)
```

Betweenness Centrality

```r
df_degree$between <- igraph::betweenness(g_df)

df_degree |>
  arrange(desc(between)) |>
  slice(1:5)
```

We can calculate the network level score of betweenness centralization

```r
centr_betw(g_df)$centralization
```

```
## [1] 0.02636484
```

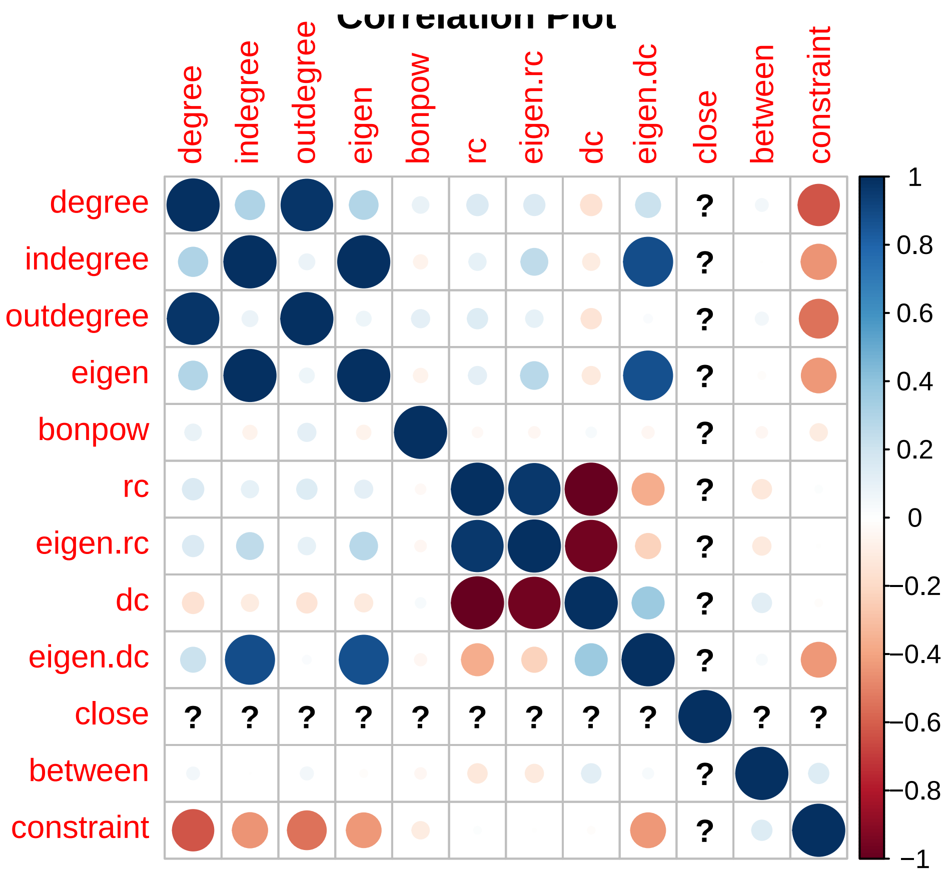
Network Constraint

```r
df_degree$constraint <- constraint(g_df)

df_degree |> arrange(constraint) |> slice(1:5)
df_degree |> arrange(desc(constraint)) |> slice(1:5)
```

Feeds & organic fertilizer, Metallic ores and Medicaments are the most reduntant industries while Misc. ceramic, stone & clay products, Glass & glass products and Industrial inorganic chemicals industries are the least reduntant ones.

Centrality Measure Correlations

```r
corrplot :: corrplot(cor(df_degree[ , -1]), title = 'Correlation Plot')
```




```
table1 <- kableExtra :: kable(apply(df_degree[ , -1], 2, function (x) df_degree$name[order(x, decreasing
table1
```

degree

indegree

outdegree

eigen

bonpow

rc

eigen.rc

dc

eigen.dc

close

between

constraint

Misc. manufacturing products

Pig iron & crude steel

Commerce

Pig iron & crude steel

Petrochemical basic products

Petroleum refinery products

Pig iron & crude steel

Feed conversion

Public administration

Misc. electronic components

Glass & glass products

Feeds & organic fertilizer

Transport & post service

Public administration

Miscellaneous metal products

Public administration

Medical service etc.

Steel products

Petroleum refinery products

Gasification

Misc. manufacturing products

Household electronics equipment

Pottery, china & earthenware

Metallic ores

Personal services

Misc. manufacturing products

Misc. manufacturing products

Personal services

Textile products

Motor vehicle parts & accessories

Motor vehicle parts & accessories

Composting

Lumber and wood products

Forestry

Forestry

Medicaments

Business services

Transport & post service

Wearing apparel etc.

Transport & post service

Non-ferrous metal products

Pig iron & crude steel

Steel products

Refuse derived fuel

Personal services

Medical service etc.

Chemical fertilizer

Livestock

Miscellaneous metal products

Personal services

Final chemical products

Misc. manufacturing products

Glass & glass products

Passenger motor cars

Passenger motor cars

Filtration

Public construction

Glass & glass products

Metallic ores

Coal mining etc.

Final chemical products

Business services

Petroleum refinery products

Business services

General-purpose machinery

Medical service etc.

Medical service etc.

Metallic ores

Misc. transportation equipment & repair

Steel products

Coal products

Miscellaneous cars

Repair of construction

Production machinery

Electricity

Production machinery

Cement & cement products

Foods

Foods

Coal mining etc.

Production machinery

Pottery, china & earthenware

Feeds & organic fertilizer

Passenger motor cars

Communications & broadcasting

Lumber and wood products

Transport & post service

Building construction

Synthetic fibers

Non-ferrous metal products

Non-ferrous metal products

Concentration

Building construction

Chemical fertilizer

Lumber and wood products

Petrochemical basic products

Activities not elsewhere classified

Building construction

Printing etc.

Misc. transportation equipment & repair

Pottery, china & earthenware

Real estate services

Transport & post service

Agricultural services

Transport & post service

Non-ferrous metal products

Textile products

Fishery

Lumber and wood products

General-purpose machinery

Repair of construction

Lumber and wood products

Metal products for construction

Communications & broadcasting

Communications & broadcasting

Pottery, china & earthenware

Ships & repair of ships

Rubber products

Pig iron & crude steel

Pig iron & crude steel

```
industry_io.node <-data.frame(apply(df_degree[ , -1], 2, function (x) df_degree$name[order(x, decreasin
industry_io.node
df_degree_io <- df_degree
```

These are the industries having the highest of centrality measures for the industries network. The nodes having higher indegrees have more inwards directed edges while outdegree gives an idea about how the outwards connections are for the node. Eigen vector, Bonacich value, Eigen Reflected Centrality, Eigen Derived Centrality are various measures of centrality of nodes. These measures suggests how influential the nodes are. Betweenness is a measure of the position of nodes in terms of closeness to other influential nodes. Constraint tells us about the level of redundancy of a node in the network to create connections with other neighbouring nodes.

Most and Least influential industries and their contribution to the waste output

Eventhough the different measures of centrality talks about the significance and influence of nodes, we are taking into consideration, Bonacich power and Constraint here to compare the waste output.

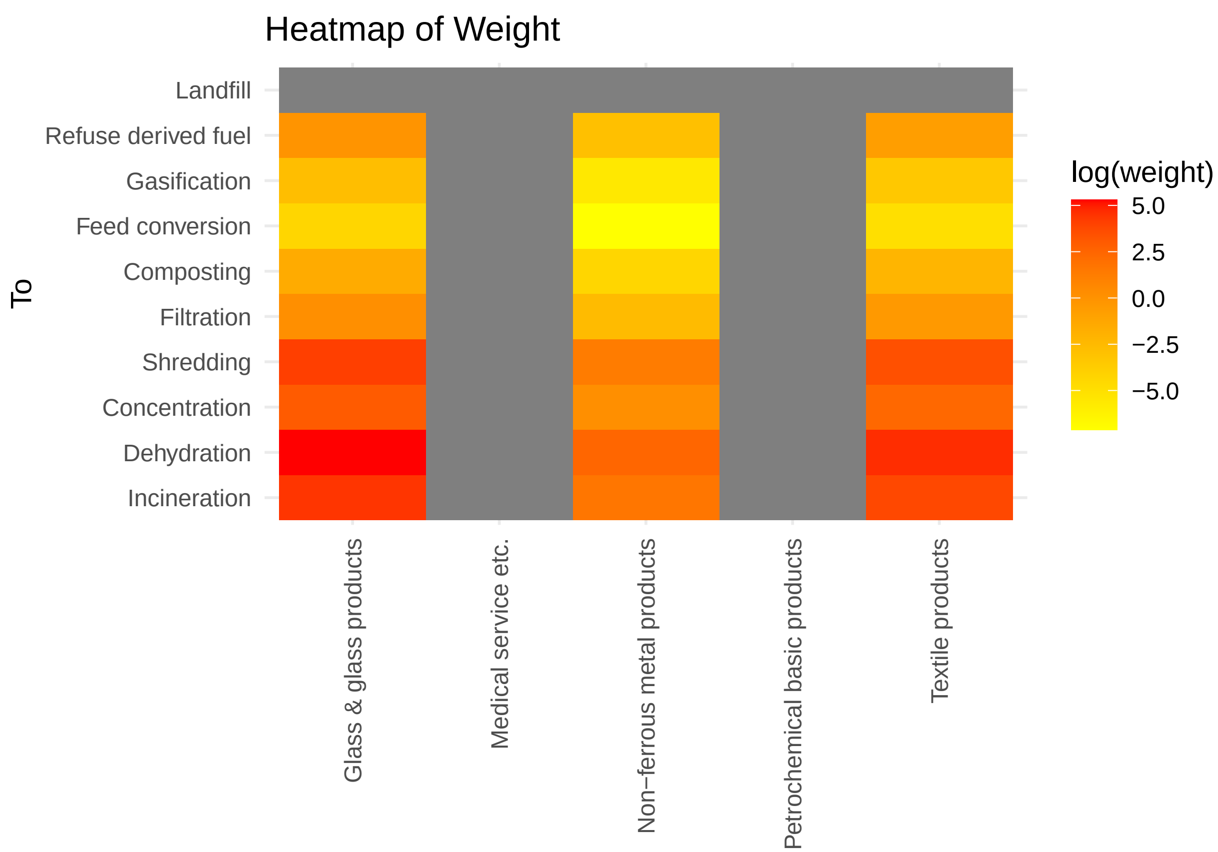
```
df_bonpow <- df_degree |>
  arrange(desc(bonpow)) |>
  slice(1:5)

df_constraint <- df_degree |> arrange(desc(constraint)) |> slice(1:5)

industry_waste <- data_2011 |> select(
...1, 83:92)
industry_waste

filtered_data1 <- industry_waste[industry_waste$...1 %in% df_bonpow$name, ]
filtered_data2 <- industry_waste[industry_waste$...1 %in% df_constraint$name, ]
```

```
combined_df_long <- melt(combined_df, id.vars = "...1", variable.name = "to", value.name = "weight", va

# Rename the "Industries" column to "From"
colnames(combined_df_long)[colnames(combined_df_long) == "...1"] <- "from"
combined_df_long
```

Waste output from most significant industries

```
filter1_long <- melt(filtered_data1, id.vars = "...1", variable.name = "to", value.name = "weight", var
```

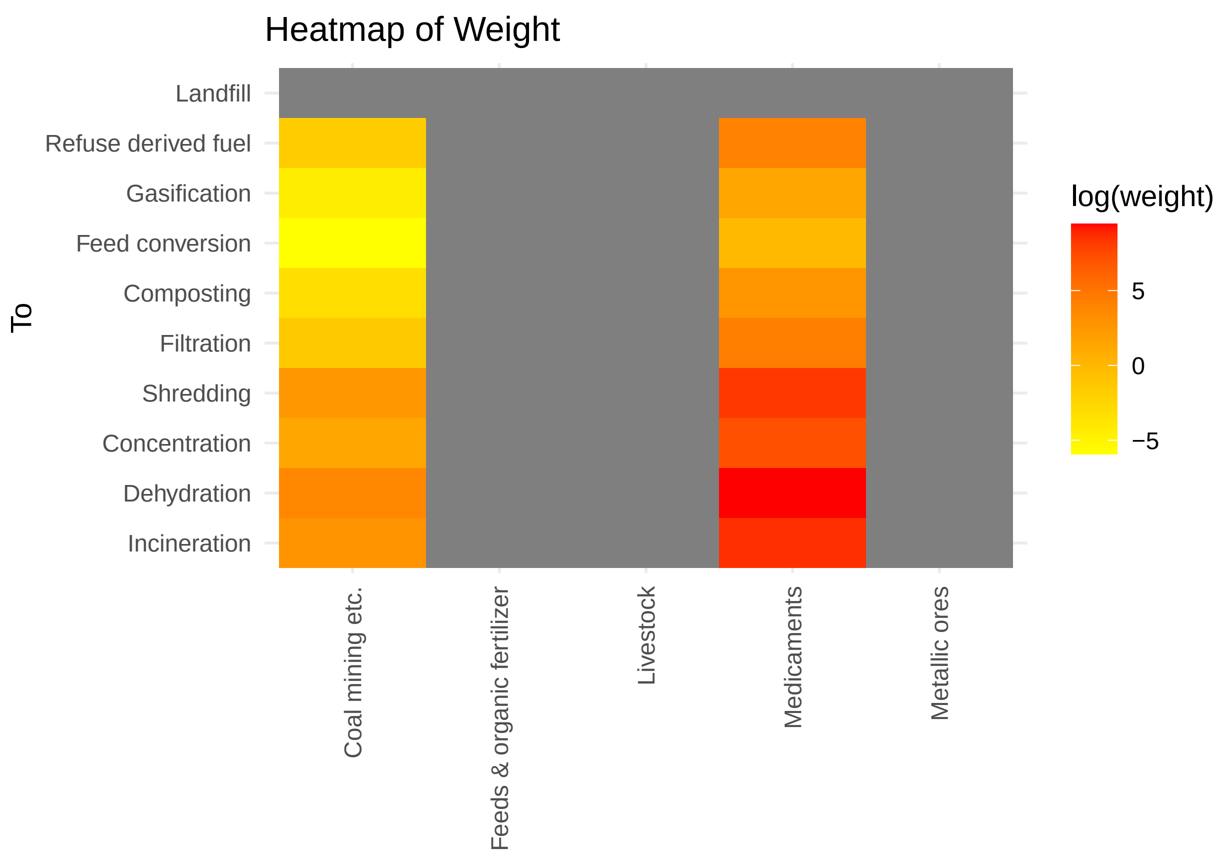
```r
# Rename the "Industries" column to "From"
colnames(filter1_long)[colnames(filter1_long) == "...1"] <- "from"
#filter1_long


# Create a heatmap plot
ggplot(filter1_long, aes(x = from, y = to, fill = log(weight))) +
  geom_tile() +
  scale_fill_gradient(low = "yellow", high = "red") +  # Change the color pattern
  labs(x = NULL, y = "To", title = "Heatmap of Weight") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))  # Rotate x-axis labels
```



Waste output from industries with high redundancy

```r
filter2_long <- melt(filtered_data2, id.vars = "...1", variable.name = "to", value.name = "weight", vari

# Rename the "Industries" column to "From"
colnames(filter2_long)[colnames(filter2_long) == "...1"] <- "from"
#filter2_long


# Create a heatmap plot
ggplot(filter2_long, aes(x = from, y = to, fill = log(weight))) +
  geom_tile() +
  scale_fill_gradient(low = "yellow", high = "red") +  # Change the color pattern
  labs(x = NULL, y = "To", title = "Heatmap of Weight") +
```

```
theme_minimal() +
theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust = 1))  # Rotate x-axis labels
```



Conclusion

The results show how much wastes go to each of the waste processing methods from the industries having most centrality measures and the most redundancy measures.

Limitation

The dataset is an older data. It is from Japan. Taking these things into consideration, we cannot make a solid conclusion about the waste outputs from industries in other parts of the world today. Also, we can compare the weight outputs from industries having each of the centrality score high and low. This would make the analysis even harder. So this study has been concluded with the available results.

Reference:

1) Nakamura, S. (2020). Tracking the Product Origins of Waste for Treatment Using the WIO Data Developed by the Japanese Ministry of the Environment. Env. Sci. Technol. https://doi.org/10.1021/acs.est.0c06015