Take-Home\_Ex02

Liaw Ying Ting, Celin

2025-05-27

# Overview - Mini Challenge 3

For this take-home exercise 2, Mini-Challenge 3 will be chosen and the visualisation will be done on this topic. The background of the mini challenge and the questions will be listed below. For this mini challenge, I will only be displaying and addressing **Questions 2 and 3** below.

# Background

Over the past decade, the community of Oceanus has faced numerous transformations and challenges evolving from its fishing-centric origins. Following major crackdowns on illegal fishing activities, suspects have shifted investments into more regulated sectors such as the ocean tourism industry, resulting in growing tensions. This increased tourism has recently attracted the likes of international pop star Sailor Shift, who announced plans to film a music video on the island.

Clepper Jessen, a former analyst at FishEye and now a seasoned journalist for the Hacklee Herald, has been keenly observing these rising tensions. Recently, he turned his attention towards the temporary closure of Nemo Reef. By listening to radio communications and utilizing his investigative tools, Clepper uncovered a complex web of expedited approvals and secretive logistics. These efforts revealed a story involving high-level Oceanus officials, Sailor Shift’s team, local influential families, and local conservationist group The Green Guardians, pointing towards a story of corruption and manipulation.

# Task and Question

Clepper diligently recorded all intercepted radio communications over the last two weeks. With the help of his intern, they have analyzed their content to identify important events and relationships between key players. The result is a knowledge graph describing the last two weeks on Oceanus. Clepper and his intern have spent a large amount of time generating this knowledge graph, and they would now like some assistance using it to answer the following questions.

1. **Clepper found that messages frequently came in at around the same time each day.**
   * Develop a graph-based visual analytics approach to identify any daily temporal patterns in communications.
   * How do these patterns shift over the two weeks of observations?
   * Focus on a specific entity and use this information to determine who has influence over them.
2. **Clepper has noticed that people often communicate with (or about) the same people or vessels, and that grouping them together may help with the investigation.**
   * Use visual analytics to help Clepper understand and explore the interactions and relationships between vessels and people in the knowledge graph.
   * Are there groups that are more closely associated? If so, what are the topic areas that are predominant for each group?
     + For example, these groupings could be related to: Environmentalism (known associates of Green Guardians), Sailor Shift, and fishing/leisure vessels.
3. **It was noted by Clepper’s intern that some people and vessels are using pseudonyms to communicate.**
   * Expanding upon your prior visual analytics, determine who is using pseudonyms to communicate, and what these pseudonyms are.
   * Some that Clepper has already identified include: “Boss”, and “The Lookout”, but there appear to be many more.
   * To complicate the matter, pseudonyms may be used by multiple people or vessels.
   * Describe how your visualizations make it easier for Clepper to identify common entities in the knowledge graph.
   * How does your understanding of activities change given your understanding of pseudonyms?
4. **Clepper suspects that Nadia Conti, who was formerly entangled in an illegal fishing scheme, may have continued illicit activity within Oceanus.**
   * Through visual analytics, provide evidence that Nadia is, or is not, doing something illegal.
   * Summarize Nadia’s actions visually. Are Clepper’s suspicions justified?

**Reflection Questions**

* Given the task to develop visualizations for knowledge graphs, did you find that the challenge pushed you to develop new techniques for visual representation?
* Did you participate in last year’s challenge? If so, did your experience last year help prepare you for this year’s challenge?
* What was the most difficult part of working on this year’s data and what could have made it more accessible?

# MC3 Kickstarter

## Getting Started

In the code chunk below, p\_load() of pacman package is used to load the R packages into R environment.

pacman::p\_load(tidyverse, jsonlite,  
 tidygraph, ggraph,  
 SmartEDA, lubridate,  
 dplyr, igraph,  
 grid)

## Import knowledge Graph Data

For this exercise, *mc3.json* file will be used. In the code chunk below, fromJSON() of jsonlite package is used to import *mc3.json* file into R and save the output object.

MC3 <- fromJSON("data/MC3\_graph.json")  
MC3\_schema <- fromJSON("data/MC3\_schema.json")

## Inspecting Knowledge graph structure

In the code chunk below, glimpse() is used to reveal the structure of the *mc3* knowledge graph.

glimpse(MC3)

## Extracting the edges and nodes tables

Next, as\_tibble() of tibble package is used to extract the nodes and links tibble data frames from *mc3* tibble dataframe into two separate tibble dataframes called *mc3\_nodes* and *mc3\_edges* respectively.

mc3\_nodes <- as\_tibble(MC3$nodes)  
mc3\_edges <- as\_tibble(MC3$edges)

## Initial EDA

In the code chunk below, ExpCatViz() of SmartEDA package is used to reveal the frequency distribution of all categorical fields in *mc3\_nodes* tibble dataframe

ExpCatViz(data=mc3\_nodes,  
 col="lightblue")

On the other hand, code chunk below uses ExpCATViz() of SmartEDA package to reveal the frequency distribution of all categorical fields in *mc3\_edges* tibble dataframe

ExpCatViz(data=mc3\_edges,  
 col="lightblue")

## Data Cleaning and Wrangling

The code chunk below performs the following data cleaning tasks:

* convert values in id field into character data type
* exclude records with id value
* exclude records with similar id values
* exclude thing\_collected field
* save the cleaned tibble dataframe into a new tibble datatable called mc\_nodes\_cleaned

mc3\_nodes\_cleaned <- mc3\_nodes %>%  
 mutate(id = as.character(id)) %>%  
 filter(!is.na(id)) %>%  
 distinct(id, .keep\_all = TRUE) %>%  
 select(-thing\_collected)

## Cleaning and wrangling edges

Next, the code chunk will be used to:

* rename source and target fields to form\_id and to\_id respectively
* convert values in from\_id and to\_id fields to character data type
* exclude values in from\_id and to\_id which is not found in the id field of mc3\_nodes\_cleaned
* exclude records whereby from\_id and/or to\_id values are missing
* save the cleaned tibble dataframe and call it mc3\_edges\_cleaned

mc3\_edges\_cleaned <- mc3\_edges %>%  
 rename(from\_id = source,  
 to\_id = target) %>%  
 mutate(across(c(from\_id, to\_id),  
 as.character)) %>%  
 filter(from\_id %in% mc3\_nodes\_cleaned$id,  
 to\_id %in% mc3\_nodes\_cleaned$id) %>%  
 filter(!is.na(from\_id), !is.na(to\_id))

Next, code chunk below will be used to create mapping of character id in mc3\_nodes\_cleaned to row index.

node\_index\_lookup <- mc3\_nodes\_cleaned %>%  
 mutate(.row\_id = row\_number()) %>%  
 select(id, .row\_id)

Next, the code chunk below will be used to join and convert from\_id and to\_id to integer indices. At the same time, we will drop rows with unmatched nodes.

mc3\_edges\_indexed <- mc3\_edges\_cleaned %>%  
 left\_join(node\_index\_lookup,   
 by = c("from\_id" = "id")) %>%  
 rename(from = .row\_id) %>%  
 left\_join(node\_index\_lookup,   
 by = c("to\_id" = "id")) %>%  
 rename(to = .row\_id) %>%  
 select(from, to, is\_inferred, type) %>%  
 filter(!is.na(from) & !is.na(to))

Next, the code chunk below is used to subset nodes to only those referenced by edges

used\_node\_indices <- sort(  
 unique(c(mc3\_edges\_indexed$from,   
 mc3\_edges\_indexed$to)))  
  
mc3\_nodes\_final <- mc3\_nodes\_cleaned %>%  
 slice(used\_node\_indices) %>%  
 mutate(new\_index = row\_number())

We will then use th code chunk below to rebuild lookup from old index to new index

old\_to\_new\_index <- tibble(  
 old\_index = used\_node\_indices,  
 new\_index = seq\_along(  
 used\_node\_indices))

Lastly, the code chunk below will be used to update edge indices to match new node table

mc3\_edges\_final <- mc3\_edges\_indexed %>%  
 left\_join(old\_to\_new\_index,   
 by = c("from" = "old\_index")) %>%  
 rename(from\_new = new\_index) %>%  
 left\_join(old\_to\_new\_index,   
 by = c("to" = "old\_index")) %>%  
 rename(to\_new = new\_index) %>%  
 select(from = from\_new, to = to\_new,   
 is\_inferred, type)

glimpse(mc3\_edges\_final)

# Building the tidygraph object

mc3\_graph <- tbl\_graph(  
 nodes = mc3\_nodes\_final,  
 edges = mc3\_edges\_final,  
 directed = TRUE  
)

str(mc3\_graph)

# Visualising the knowledge graph

Several of the ggraph layouts involve randomisation. In order to ensure reproducibility, it is necessary to set the seed value before plotting by using the code chunk below.

set.seed(1234)

In the code chunk below, ggraph functions are used to create the whole graph

ggraph(mc3\_graph,   
 layout = "fr") +  
 geom\_edge\_link(alpha = 0.3,   
 colour = "gray") +  
 geom\_node\_point(aes(color = `type`),   
 size = 4) +  
 geom\_node\_text(aes(label = type),   
 repel = TRUE,   
 size = 1.0) +  
 theme\_void()

# Question 2

### Data Cleaning

For nodes, the code filter() will be used to filter out only *Person and Vessel* from *Entity*, and *Communication* from *Event* and their *relationship.*

After this filter, the nodes only contain:

* entity nodes whose sub\_type is Person or Vessel
* Event nodes whose sub\_type is Communication
* Any node whose type is Relationship.

mutate() adds a new column called idx to the filtered data frame

select() extracts 2 columns from *selected\_nodes* and give them a new id and rename to *id\_map*

For Edges, filter() keeps only those where both endpoints are in the *old* column of *id\_map.*

left\_join() matches each row’s *to* value against *id\_map$old* and brings in *id\_map$new* for that *to* node and renames it *to\_new*. Same thing for the *from* column.

selected\_nodes <- mc3\_nodes\_final %>%  
 filter(  
 (type == "Entity" & sub\_type %in% c("Person", "Vessel")) |  
 (type == "Event" & sub\_type == "Communication") |  
 (type == "Relationship")  
 ) %>%  
 mutate(idx = row\_number())  
  
# mapping old\_index → new idx  
id\_map <- selected\_nodes %>%  
 select(old = new\_index, new = idx)  
  
# 2. filter edges so both ends are in selected\_nodes  
selected\_edges <- mc3\_edges\_final %>%  
 filter(from %in% id\_map$old, to %in% id\_map$old) %>%  
   
 # 3. re-index the endpoints  
 left\_join(id\_map, by = c("from" = "old")) %>%  
 rename(from\_new = new) %>%  
 left\_join(id\_map, by = c("to" = "old")) %>%  
 rename(to\_new = new) %>%  
 select(from = from\_new, to = to\_new, type)

### People/Vessel Vs Relationship Subtypes

Step 1: we proceed to filter() only the edges that we want, specifically *sent, received and evidence\_for*. The labels for the edges show “sent”, “received” and “evidence\_for” using transmute().

Step 2: We then build the node table and colour by sub\_type which is called *group*. The label for the names/vessels will be placed under the node using *label.*

Step 3: Draw a Visnetwork graph.

library(dplyr)  
library(tidyr) # for pivot\_longer()  
library(visNetwork)  
library(RColorBrewer)  
  
# 1. Build sender/receiver pairs and their relationship nodes  
send\_pairs <- mc3\_edges\_final %>%  
 filter(type == "sent") %>%  
 rename(sender = from, comm = to) %>%  
 select(sender, comm)  
  
recv\_pairs <- mc3\_edges\_final %>%  
 filter(type == "received") %>%  
 rename(receiver = to, comm = from) %>%  
 select(receiver, comm)  
  
rel\_edges <- mc3\_edges\_final %>%  
 filter(type %in% c("evidence\_for", "is\_inferred")) %>%  
 rename(rel = to, comm = from) %>%  
 select(rel, comm)  
  
# 2. Join to get each (sender, receiver, rel) for every communication event  
triplets <- send\_pairs %>%  
 inner\_join(recv\_pairs, by = "comm") %>%  
 inner\_join(rel\_edges, by = "comm") %>%  
 mutate(  
 person1 = pmin(sender, receiver),  
 person2 = pmax(sender, receiver)  
 ) %>%  
 distinct(person1, person2, rel, .keep\_all = TRUE) %>%  
 select(person1, person2, rel)  
  
# 3. Collect unique node‐IDs used (persons/vessels and relationship nodes)  
node\_ids\_used <- unique(c(triplets$person1, triplets$person2, triplets$rel))  
  
# 4. Build a lookup of node info for those IDs,  
# but filter to only Person/Vessel entity subtypes and Relationship nodes  
nodes\_sub <- mc3\_nodes\_final %>%  
 filter(new\_index %in% node\_ids\_used) %>%  
 filter(  
 sub\_type %in% c("Person", "Vessel") |  
 type == "Relationship"  
 ) %>%  
 select(new\_index, label, sub\_type, type)  
  
# 4a. Define pleasing, non‐grey colors for entity subtypes  
entity\_color\_map <- tibble(  
 sub\_type = c("Person", "Vessel"),  
 color = c("#2c7fb8", "#e6550d") # blue for Person, orange for Vessel  
)  
  
# 4b. Identify all relationship subtypes and assign distinct, visually pleasing colors  
rel\_subtypes <- nodes\_sub %>%  
 filter(type == "Relationship") %>%  
 distinct(sub\_type) %>%  
 arrange(sub\_type) %>%  
 pull(sub\_type)  
  
n\_rels <- length(rel\_subtypes)  
if (n\_rels > 0) {  
 base\_palette <- brewer.pal(min(n\_rels, 8), "Set2")  
 if (n\_rels > length(base\_palette)) {  
 rel\_colors <- colorRampPalette(base\_palette)(n\_rels)  
 } else {  
 rel\_colors <- base\_palette[1:n\_rels]  
 }  
 rel\_color\_map <- tibble(  
 sub\_type = rel\_subtypes,  
 color = rel\_colors  
 )  
} else {  
 rel\_color\_map <- tibble(sub\_type = character(0), color = character(0))  
}  
  
# 4c. Combine entity and relationship color mappings  
color\_map <- bind\_rows(entity\_color\_map, rel\_color\_map)  
  
# 4d. Build nodes\_vis with assigned colors (no grey)  
nodes\_vis <- nodes\_sub %>%  
 left\_join(color\_map, by = "sub\_type") %>%  
 transmute(  
 id = new\_index,  
 label = label,  
 sub\_type = sub\_type,  
 color = color  
 )  
  
# 5. Build the visNetwork 'edges' data frame  
# 5a. Edges between the two people/vessels (no label)  
edges\_comm\_vis <- triplets %>%  
 transmute(  
 from = person1,  
 to = person2,  
 label = "", # no “comm” text  
 width = 2,  
 arrows = "to"  
 )  
  
# 5b. For each triplet, connect both persons to the relationship node (no label)  
edges\_rel\_vis <- triplets %>%  
 pivot\_longer(  
 cols = c(person1, person2),  
 names\_to = NULL,  
 values\_to = "person"  
 ) %>%  
 transmute(  
 from = person,  
 to = rel,  
 label = "", # no text here  
 width = 1,  
 arrows = "to"  
 )  
  
# 5c. Combine both sets of edges  
edges\_vis <- bind\_rows(edges\_comm\_vis, edges\_rel\_vis)  
  
# 6. Draw the visNetwork with a dropdown filter by sub\_type.  
# Legend shows only subtypes in the graph, in oval shape, larger, and using our custom colors.  
visNetwork(nodes\_vis, edges\_vis) %>%  
 visNodes(  
 borderWidth = 1,  
 borderWidthSelected = 2  
 ) %>%  
 visEdges(  
 arrows = "to",  
 color = list(color = "#333333", highlight = "#FF0000"), # dark edge color  
 font = list(align = "middle")  
 ) %>%  
 visOptions(  
 highlightNearest = TRUE,  
 nodesIdSelection = TRUE,  
 selectedBy = "sub\_type"  
 ) %>%  
 visLegend(  
 addNodes = data.frame(  
 label = color\_map$sub\_type,  
 shape = "ellipse",   
 color = color\_map$color,  
 size = rep(20, nrow(color\_map))  
 ),  
 useGroups = FALSE  
 )

### People vs Relationship Subtype

# 7. Exclude all “Vessel” nodes and any edges involving them  
nodes\_vis2 <- nodes\_vis %>%  
 filter(sub\_type != "Vessel")  
  
edges\_vis2 <- edges\_vis %>%  
 filter(from %in% nodes\_vis2$id, to %in% nodes\_vis2$id)  
  
# 8. Build a reduced legend data.frame for the remaining subtypes  
legend\_map2 <- color\_map %>%  
 filter(sub\_type %in% unique(nodes\_vis2$sub\_type))  
  
# 9. Draw the new visNetwork without Vessel nodes  
visNetwork(nodes\_vis2, edges\_vis2) %>%  
 visNodes(  
 borderWidth = 1,  
 borderWidthSelected = 2  
 ) %>%  
 visEdges(  
 arrows = "to",  
 color = list(color = "#333333", highlight = "#FF0000"),  
 font = list(align = "middle")  
 ) %>%  
 visOptions(  
 highlightNearest = TRUE,  
 nodesIdSelection = TRUE,  
 selectedBy = "sub\_type"  
 ) %>%  
 visLegend(  
 addNodes = data.frame(  
 label = legend\_map2$sub\_type,  
 shape = "ellipse",  
 color = legend\_map2$color,  
 size = rep(20, nrow(legend\_map2))  
 ),  
 useGroups = FALSE  
 )

### Q2 Part A

By selecting the different sub-groups from the drop-down below, we can conclude the following information.

**Colleagues:**

* Boss, Mrs Money and The Middleman
* Boss, The Accountant and The Middleman
* The Lookout, The Intern and Mrs Money
* Mrs Money and The Intern
* Boss and Mrs Money
* Boss and Rodriguez
* Small Fry and Davis
* Nadia Conti and Davis
* Nadia Conti and Elise
* Nadia Conti and Liam Thorne
* Sam and Kelly

**Friends:**

* Sam and Kelly
* The Intern and The Lookout

**Unfriendly:**

* Elise and Nadia Conti

**Suspicious:**

* Boss, Mrs Money, The Intern, The Lookout, The Accountant
* Sam, Kelly, Liam Thorne, Elise, Nadia Conti, Davis

From the above results, it can be seen that either Sam or Kelly holds the status of The Intern and The Lookout as they are the only ones with the relationship of friends in the conclusion above.

Boss seems to be the one with the most information and power as Mrs Money, The Accountant and The Middlman transmit information to The Boss only

## Q2 Part B

Based on the question 2b, I have categorised them into the 3 different Entity Subgroups: Person, Organisation and Vessel

### Group by organisation

library(dplyr)  
library(tidytext)  
library(wordcloud)  
  
# 1. Extract communication‐event nodes with their new\_index and content  
comm\_nodes <- mc3\_nodes\_final %>%  
 filter(type == "Event", sub\_type == "Communication") %>%  
 select(event\_index = new\_index, content)  
  
# 2. Build a mapping from communication events to entities (Organization subtype)  
send\_map <- mc3\_edges\_final %>%  
 filter(type == "sent") %>%  
 transmute(entity\_index = from, event\_index = to)  
  
recv\_map <- mc3\_edges\_final %>%  
 filter(type == "received") %>%  
 transmute(event\_index = from, entity\_index = to)  
  
event\_to\_entity <- bind\_rows(send\_map, recv\_map)  
  
# 3. Join event\_to\_entity with comm\_nodes, then keep only Organization entities  
comm\_by\_entity <- event\_to\_entity %>%  
 inner\_join(comm\_nodes, by = "event\_index") %>%  
 inner\_join(  
 mc3\_nodes\_final %>%   
 select(entity\_index = new\_index, sub\_type, label),  
 by = "entity\_index"  
 ) %>%  
 filter(sub\_type == "Organization") %>%  
 select(organization = label, content)  
  
# 4. Collapse all texts per organization  
org\_texts <- comm\_by\_entity %>%  
 group\_by(organization) %>%  
 summarize(text = paste(content, collapse = " "), .groups = "drop")  
  
# 5. Tokenize, remove stop‐words, and compute word frequencies per organization  
tidy\_org\_words <- org\_texts %>%  
 unnest\_tokens(word, text) %>%  
 anti\_join(stop\_words, by = "word") %>%  
 count(organization, word, sort = TRUE)  
  
# 6. Plot one word cloud per organization  
org\_list <- unique(tidy\_org\_words$organization)  
n\_orgs <- length(org\_list)  
n\_cols <- 3  
n\_rows <- ceiling(n\_orgs / n\_cols)  
  
par(mfrow = c(n\_rows, n\_cols), mar = c(0, 0, 0, 0))  
for (org in org\_list) {  
 freq\_tbl <- tidy\_org\_words %>% filter(organization == org)  
 if (nrow(freq\_tbl) == 0) {  
 plot.new()  
 title(main = org)  
 next  
 }  
 v <- setNames(freq\_tbl$n, freq\_tbl$word)  
 wordcloud(  
 words = names(v),  
 freq = as.numeric(v),  
 max.words = 20,  
 scale = c(3, 0.5),  
 random.order = FALSE  
 )  
 title(main = org, line = -1)  
}  
par(mfrow = c(1, 1))

For Organisation: It can be seen that Team Glitters does not have any pre-dominant topic area.

Oceanus City Council, Sailor Shift Team and Green Guardians seems to have common words like nemo, city, reef ,equipment and vessel, showing that they are more concerned about the environment.

V. Miesel Shipping however, is more concerned about regulations as the most commonly used words are permit, operations and documentation.

### Group by Person

# (Assuming you already have mc3\_nodes\_final and mc3\_edges\_final loaded)  
  
library(dplyr)  
library(tidytext)  
library(wordcloud)  
  
# 1. Extract communication‐event nodes with their new\_index and content  
comm\_nodes <- mc3\_nodes\_final %>%  
 filter(type == "Event", sub\_type == "Communication") %>%  
 select(event\_index = new\_index, content)  
  
# 2. Build a mapping from communication events to entities  
send\_map <- mc3\_edges\_final %>%  
 filter(type == "sent") %>%  
 transmute(entity\_index = from, event\_index = to)  
  
recv\_map <- mc3\_edges\_final %>%  
 filter(type == "received") %>%  
 transmute(event\_index = from, entity\_index = to)  
  
event\_to\_entity <- bind\_rows(send\_map, recv\_map)  
  
# 3. Join event\_to\_entity with comm\_nodes, then keep only Person entities  
comm\_by\_person <- event\_to\_entity %>%  
 inner\_join(comm\_nodes, by = "event\_index") %>%  
 inner\_join(  
 mc3\_nodes\_final %>%   
 select(entity\_index = new\_index, sub\_type, label),  
 by = "entity\_index"  
 ) %>%  
 filter(sub\_type == "Person") %>%  
 select(person = label, content)  
  
# 4. Collapse all texts per person  
person\_texts <- comm\_by\_person %>%  
 group\_by(person) %>%  
 summarize(text = paste(content, collapse = " "), .groups = "drop")  
  
# 5. Tokenize, remove stop‐words, and compute word frequencies per person  
tidy\_person\_words <- person\_texts %>%  
 unnest\_tokens(word, text) %>%  
 anti\_join(stop\_words, by = "word") %>%  
 count(person, word, sort = TRUE)  
  
# 6. Plot one word cloud per person  
person\_list <- unique(tidy\_person\_words$person)  
n\_persons <- length(person\_list)  
n\_cols <- 4  
n\_rows <- ceiling(n\_persons / n\_cols)  
  
par(mfrow = c(n\_rows, n\_cols), mar = c(0, 0, 0, 0))  
for (p in person\_list) {  
 freq\_tbl <- tidy\_person\_words %>% filter(person == p)  
 if (nrow(freq\_tbl) == 0) {  
 plot.new()  
 title(main = p)  
 next  
 }  
 v <- setNames(freq\_tbl$n, freq\_tbl$word)  
 wordcloud(  
 words = names(v),  
 freq = as.numeric(v),  
 max.words = 10,  
 scale = c(3, 0.5),  
 random.order = FALSE  
 )  
 title(main = p, line = -1)  
}  
par(mfrow = c(1, 1))

**For Person Subgroups:**

**Top 3 Commonly Used Words:**

Mrs Money: Meeting, Nemo Reef, Intern

The Intern: Nemo Reef, Lookout, Money

The Lookout: Nemo Reef, Intern, tomorrow

Boss: Nemo Reef, meeting, middleman

The Middleman: Nemo Reef, boss, meeting

The Accountant: meeting, middleman, boss

**It can be seen that these 6 people are in contact with each other as their most commonly used terms concern each other and usually revolves around Nemo Reef.**

Liam Thorne: Nemo Reef, council, report

Nadia Conti: Nemo Reef, documentation, meeting

Rodriguez: Nemo reef, davis, remora

Samantha Blake: Nemo Reef, tomorrow, city council

Elise: Nemo Reef, tomorrow, meeting

**It can also be seen that these group of people have common schedules like meeting, tomorrow and City Council, and they are also concerned about Nemo Reef.**

Clepper Jensen: Miranda, equipment, environmental

Miranda Jordan: Miranda, equipment, clepper Jensen

Davis: Mako, security, equipment

Sam: birdwatching, lookout, reef

Mako: try, team, remora

Kelly: sam, kelly

**These group of people here have different focus and do not have commonailities with the groups above.**

### Group by Vessels

# 1. Extract communication‐event nodes with their new\_index and content  
comm\_nodes <- mc3\_nodes\_final %>%  
 filter(type == "Event", sub\_type == "Communication") %>%  
 select(event\_index = new\_index, content)  
  
# 2. Build a mapping from communication events to entities (sender/receiver)  
send\_map <- mc3\_edges\_final %>%  
 filter(type == "sent") %>%  
 transmute(entity\_index = from, event\_index = to)  
  
recv\_map <- mc3\_edges\_final %>%  
 filter(type == "received") %>%  
 transmute(event\_index = from, entity\_index = to)  
  
event\_to\_entity <- bind\_rows(send\_map, recv\_map)  
  
# 3. Join event\_to\_entity with comm\_nodes, then keep only Vessel entities  
comm\_by\_vessel <- event\_to\_entity %>%  
 inner\_join(comm\_nodes, by = "event\_index") %>%  
 inner\_join(  
 mc3\_nodes\_final %>%  
 select(entity\_index = new\_index, sub\_type, label),  
 by = "entity\_index"  
 ) %>%  
 filter(sub\_type == "Vessel") %>% # <— change here  
 select(vessel = label, content)  
  
# 4. Collapse all texts per vessel  
vessel\_texts <- comm\_by\_vessel %>%  
 group\_by(vessel) %>%  
 summarize(text = paste(content, collapse = " "), .groups = "drop")  
  
# 5. Tokenize, remove stop‐words, and compute word frequencies per vessel  
tidy\_vessel\_words <- vessel\_texts %>%  
 unnest\_tokens(word, text) %>%  
 anti\_join(stop\_words, by = "word") %>%  
 count(vessel, word, sort = TRUE)  
  
# 6. Plot one word cloud per vessel  
vessel\_list <- unique(tidy\_vessel\_words$vessel)  
n\_vess <- length(vessel\_list)  
n\_cols <- 4  
n\_rows <- ceiling(n\_vess / n\_cols)  
  
par(mfrow = c(n\_rows, n\_cols), mar = c(0, 0, 0, 0))  
for (ves in vessel\_list) {  
 freq\_tbl <- tidy\_vessel\_words %>% filter(vessel == ves)  
 if (nrow(freq\_tbl) == 0) {  
 plot.new()  
 title(main = ves)  
 next  
 }  
 v <- setNames(freq\_tbl$n, freq\_tbl$word)  
 wordcloud(  
 words = names(v),  
 freq = as.numeric(v),  
 max.words = 20,  
 scale = c(3, 0.5),  
 random.order = FALSE  
 )  
 title(main = ves, line = -1)  
}  
par(mfrow = c(1, 1))

# Question 3

### Daily Network Graph of Communication

library(patchwork)  
  
  
daily\_comms <- selected\_nodes %>%  
 filter(type=="Event", sub\_type=="Communication") %>%  
 mutate(date=as\_date(timestamp)) %>%  
 select(idx, date)  
  
all\_dates <- sort(unique(daily\_comms$date)) %>% discard(is.na)  
  
plots <- map(all\_dates, function(d) {  
 comm\_idx <- daily\_comms %>% filter(date==d) %>% pull(idx)  
 es <- selected\_edges %>% filter(type=="sent", to %in% comm\_idx) %>% transmute(comm=to, sender=from)  
 er <- selected\_edges %>% filter(type=="received", from %in% comm\_idx) %>% transmute(comm=from, receiver=to)  
 dp <- inner\_join(es, er, by="comm") %>% transmute(from=sender, to=receiver, type="sent")  
 if (nrow(dp)==0) return(NULL)  
  
 nodes\_d <- selected\_nodes %>%  
 filter(idx %in% unique(c(dp$from, dp$to)), sub\_type %in% c("Person", "Vessel"))  
 dp <- dp %>% filter(from %in% nodes\_d$idx, to %in% nodes\_d$idx)  
 if (nrow(dp)==0) return(NULL)  
  
 nm <- tibble(old=nodes\_d$idx, new=seq\_len(nrow(nodes\_d)))  
 edges2 <- dp %>%  
 left\_join(nm, by=c("from"="old")) %>% mutate(from=new) %>% select(-new) %>%  
 left\_join(nm, by=c("to"="old")) %>% mutate(to=new) %>% select(-new)  
 nodes2 <- nodes\_d %>% mutate(idx2=seq\_len(nrow(nodes\_d)))  
  
 g <- tbl\_graph(nodes=nodes2, edges=edges2, directed=TRUE)  
 lay <- create\_layout(g, layout="fr", niter=500, area=vcount(g)^2, repulse=vcount(g)^1.2)  
  
 ggraph(lay) +  
 geom\_edge\_link(aes(color=type),  
 arrow=arrow(length=unit(4,"mm"), type="closed"),  
 alpha=0.8, width=1) +  
 geom\_node\_point(aes(color=sub\_type), size=4) +  
 geom\_node\_text(aes(label=ifelse(sub\_type %in% c("Person","Vessel"), label, "")),  
 repel=TRUE, point.padding=unit(0.5,"lines"),  
 box.padding=unit(0.5,"lines"), size=6) +  
 scale\_edge\_color\_manual(values=c(sent="#7570B3")) +  
 scale\_color\_manual(values=c(Person="#1b9e77", Vessel="#d95f02")) +  
 ggtitle(as.character(d)) +  
 theme\_void() +  
 theme(plot.title=element\_text(size=12,hjust=0.5), legend.position="right")  
})  
  
wrap\_plots(plots, ncol=2) +  
 plot\_layout(guides="collect") +  
 plot\_annotation(title="MC3 Communications by Day")

### Communication between People and Vessels

**Additional visualisations are done below to further investigate the communication between the pseudonyms and real names.**

library(dplyr)  
library(visNetwork)  
  
# 1. Keep only Person/Vessel nodes, assigning colors by sub\_type  
nodes <- selected\_nodes %>%  
 filter(sub\_type %in% c("Person", "Vessel")) %>%  
 transmute(  
 id = idx,  
 label = label,  
 color = ifelse(sub\_type == "Person", "#1b9e77", "#d95f02")  
 )  
  
# 2. Build direct sender→receiver pairs from communication events  
es <- selected\_edges %>%  
 filter(type == "sent") %>%  
 transmute(comm = to, sender = from)  
er <- selected\_edges %>%  
 filter(type == "received") %>%  
 transmute(comm = from, receiver = to)  
  
# 3. Join on the communication node to get from→to, then count occurrences  
dp <- inner\_join(es, er, by = "comm") %>%  
 transmute(from = sender, to = receiver)  
  
edges <- dp %>%  
 count(from, to, name = "count") %>%  
 filter(from %in% nodes$id, to %in% nodes$id) %>%  
 mutate(width = count, label = as.character(count))  
  
# 4. Draw a visNetwork: node colors by type, edge width and label by count  
visNetwork(nodes, edges) %>%  
 visIgraphLayout(layout = "layout\_with\_fr", randomSeed = 123) %>%  
 visEdges(  
 arrows = "to",  
 color = list(color = "#AAAAAA", highlight = "#FF0000"),  
 font = list(align = "middle")  
 ) %>%  
 visOptions(  
 highlightNearest = TRUE,  
 nodesIdSelection = TRUE  
 )

## Q3 Part A

The pseudonyms used are:

* Boss
* The Intern
* Mrs Money
* The Lookout
* The Accountant
* The Middleman

## Q3 Part B

My visualisations help to break down the communication on a daily basis and to draw the connections between the different people based on their communication groups.

From the figures above, it can be seen that the Boss is mostly always on the receiving end of the communication, with the exception of Mako(Vessel). The ones who are in direct contact with Boss are Mrs Money, The Middleman, Rodriguez, Davis, Liam Thorne, The Accountant and Mako (Vessel).

Mrs Money, however, receives information mainly from The Intern.

The Lookout and The Intern, however, usually communicates in both directions, with The Intern usually on the receiving end of information.

The Accountant only communicated with Boss on 11 October and he is often communicating with The Middleman.

The Lookout also sends informations to Sam.

**There is a clear hierachy when sending information:**

Lookout -> Intern -> Mrs Money/The Accountant/The Middleman -> Boss

Lookout -> Sam

## Q3 Part C

These visualisations have confirmed my understanding that Boss is likely the leader of the group and the Vessel that is in contact with boss is likely Mako for illegal activities. The illegal activities should be taking place around Nemo Reef or meetings are held around Nemo Reef. The ones of higher positions are probably Mrs Money, The Middleman and The Accountant who reports directly to Boss.

Rodriguez, Liam Thorne and Davis seem to be involved directly with The Boss as well.

Strangely, Nadia Conti seems to be at the receiving end of the communciation from Elise, Davis and Rodriguez and only replies to Liam Thorne. Nadia Conti is also in contact with several vessels.

The Intern and The Lookout seems to be of the same position in the syndicate, but The Intern is the only one who communicates with Mrs Money.

The Lookout seems to be reporting to Sam as well, as the communication is always in 1 direction.

**From all the previous analysis, I would like to conclude that Rodriguez and Davis are likely involved in the illegal activities as they are in direct contact with Boss. Sam is also likely involved as The Lookout reports to him frequently.**

# References

* [ISSS608 AY2024-25 April Term](https://isss608-ay2024-25apr.netlify.app/)
* [visNetwork](https://datastorm-open.github.io/visNetwork/)