Analysis of matrix spec bias-rumors

MTRNord

The following data is based purely on public knowledge. This means data is fetched from github and gitlab as best as I was able to.

Fetching the MSCs

```
# Setup things
library(gh)
library(tidyverse)
library(ggplot2)

# Set theme
theme_set(theme_bw())

# gh_whoami()
```

```
cleanup_data <- function(prs_gql) {
  prs_gql <- prs_gql[[1]][[1]][[1]][[2]]
  for (i in 1:length(prs_gql)) {
    prs_gql[[i]] <- prs_gql[[i]]$node
    prs_gql[[i]]$temp_labels <- NA
    if (length(prs_gql[[i]]$labels$edges) >= 1) {
      prs_gql[[i]]$temp_labels <- list()
      for (y in 1:length(prs_gql[[i]]$labels$edges)) {
        prs_gql[[i]]$temp_labels[[y]] <- prs_gql[[i]]$labels$edges[[y]]$node
      }

      prs_gql[[i]]$temp_labels <- map(prs_gql[[i]]$temp_labels, as.data.frame)
      prs_gql[[i]]$temp_labels <- do.call(rbind, prs_gql[[i]]$temp_labels)
    }
    prs_gql[[i]]$labels <- NA
    prs_gql[[i]]$labels <- prs_gql[[i]]$temp_labels
    prs_gql[[i]]$temp_labels <- NULL
  }
  prs_gql <- as.data.frame(do.call(rbind, prs_gql))
  if (!("mergedAt" %in% colnames(prs_gql))) {
    prs_gql$mergedAt <- NA
    prs_gql$isPR <- FALSE
  } else {
    prs_gql$isPR <- TRUE
  }
```

```
  for (i in rownames(prs_gql)) {
    author <- prs_gql[i, "author"]
    if (!is.null(author)) {
      prs_gql[i, "author"] <- do.call(rbind.data.frame, author)
    }
  }

  return(prs_gql)
}
```

```
if (!exists("issues_gql_all")) {
  issue_query <- 'query($after: String) {
  repository(owner: "matrix-org", name: "matrix-spec-proposals") {
    issues(
      states: [OPEN, CLOSED]
      orderBy: {field: CREATED_AT, direction: ASC}
      first: 100
      after: $after
    ) {
      pageInfo {
        startCursor
        endCursor
        hasNextPage
        hasPreviousPage
      }
      edges {
        node {
          title
          url
          author {
            login
          }
          closedAt
          createdAt
          labels(first: 100) {
            pageInfo {
              startCursor
              endCursor
              hasNextPage
              hasPreviousPage
            }
            edges {
              node {
                name
                createdAt
              }
            }
          }
        }
      }
    }
  }
}'
```

```
  pr_query <- 'query($after: String) {
  repository(owner: "matrix-org", name: "matrix-spec-proposals") {
    pullRequests(
      states: [OPEN, CLOSED, MERGED]
      orderBy: {field: CREATED_AT, direction: ASC}
      first: 100
      after: $after
    ) {
      pageInfo {
        startCursor
        endCursor
        hasNextPage
        hasPreviousPage
      }
      edges {
        node {
          title
          url
          author {
            login
          }
          closedAt
          mergedAt
          createdAt
          labels(first: 100) {
            pageInfo {
              startCursor
              endCursor
              hasNextPage
              hasPreviousPage
            }
            edges {
              node {
                name
                createdAt
              }
            }
          }
        }
      }
    }
  }
}'

  issues_gql <- gh_gql(issue_query)
  issues_gql_pageinfo <- issues_gql[[1]][[1]][[1]][[1]]
  issues_gql <- cleanup_data(issues_gql)

  gql_data <- list(issues_gql)

  # Paginate API
  while (issues_gql_pageinfo$hasNextPage) {
    variables <- list()
```

```
    variables$after <- issues_gql_pageinfo$endCursor
    issues_gql <- gh_gql(issue_query, variables = variables)
    issues_gql_pageinfo <- issues_gql[[1]][[1]][[1]][[1]]
    issues_gql <- cleanup_data(issues_gql)
    gql_data[[length(gql_data) + 1]] <- issues_gql
  }

  prs_gql <- gh_gql(pr_query)
  prs_gql_pageinfo <- prs_gql[[1]][[1]][[1]][[1]]
  prs_gql <- cleanup_data(prs_gql)
  gql_data[[length(gql_data) + 1]] <- prs_gql

  # Paginate API
  while (prs_gql_pageinfo$hasNextPage) {
    variables <- list()
    variables$after <- prs_gql_pageinfo$endCursor
    prs_gql <- gh_gql(pr_query, variables = variables)
    prs_gql_pageinfo <- prs_gql[[1]][[1]][[1]][[1]]
    prs_gql <- cleanup_data(prs_gql)
    gql_data[[length(gql_data) + 1]] <- prs_gql
  }

  issues_gql_all <- do.call(rbind, gql_data)

  # Cleanup
  rm(issues_gql, prs_gql, gql_data, variables, issues_gql_pageinfo, prs_gql_pageinfo)


  issues_gql_all <- issues_gql_all |>
    rowwise()
}
```

Get Employee association from Github and Gitlab

Please note that in the current PDF this is not yet hooked up to gitlab or checking the github workplace field. It may also exclude some users that are not detectable.

```
# TODO also check against gitlab
# TODO also check workplace thingy

# Compile a list of who is who
if (!exists("element_employee") || !exists("famedly_employee") || !exists("beeper_employee")) {
  element_employee <- list()
  sct_employee <- c("ara4n", "erikjohnston", "richvdh", "dbkr", "uhoreg", "anoadragon453", "turt2live",
  famedly_employee <- list("deepbluev7", "Sorunome", "MTRNord")
  beeper_employee <- list("Fizzadar")
  users <- list()
  # Get orgs of users on github
  for (i in rownames(issues_gql_all)) {
    user <- issues_gql_all[i, "author"]
    user <- paste(unlist(user), collapse = "")
    if (is.na(user) || is.null(user) || user == "") {
```

```
      next
    }
    if ((user %in% users) || (user %in% sct_employee)) {
      next
    }
    orgs_raw <- gh(sprintf("GET /users/%s/orgs", user))
    orgs <- as.data.frame(do.call(rbind, orgs_raw))

    if ("vector-im" %in% orgs$login) {
      element_employee[[length(element_employee) + 1]] <- user
    } else if ("beeper" %in% orgs$login) {
      beeper_employee[[length(beeper_employee) + 1]] <- user
    } else if ("Famedly" %in% orgs$login) {
      famedly_employee[[length(famedly_employee) + 1]] <- user
    }
    users[[length(users) + 1]] <- user
  }
  rm(orgs, orgs_raw, user, author, i, users)
}
```

Get times for state transitions

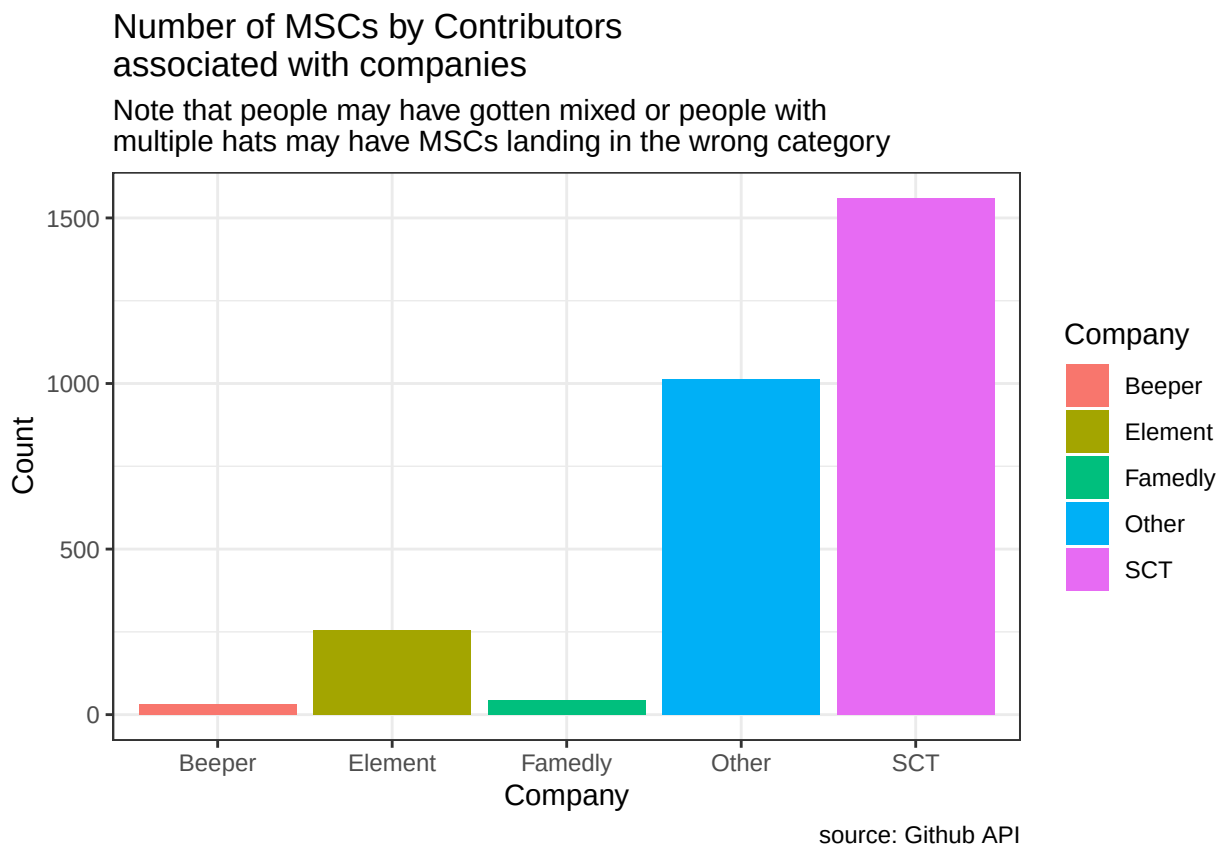
```
opened_to_proposal <- issues_gql_all
```

MSCs by Company (all kind)

Note that this does not adjust for private vs company MSCs.

```
# Filter MSCs by company
mscs_element <- issues_gql_all |>
  filter(!is.null(author) && is.element(author, element_employee)) |>
  nrow()

mscs_sct <- issues_gql_all |>
  filter(!is.null(author) && is.element(author, sct_employee)) |>
  nrow()

mscs_famedly <- issues_gql_all |>
  filter(!is.null(author) && is.element(author, famedly_employee)) |>
  nrow()

mscs_beeper <- issues_gql_all |>
  filter(!is.null(author) && is.element(author, beeper_employee)) |>
  nrow()

mscs_other <- nrow(issues_gql_all) - mscs_element - mscs_beeper - mscs_famedly - mscs_sct

column_names <- c("Count")
# Display Data
```

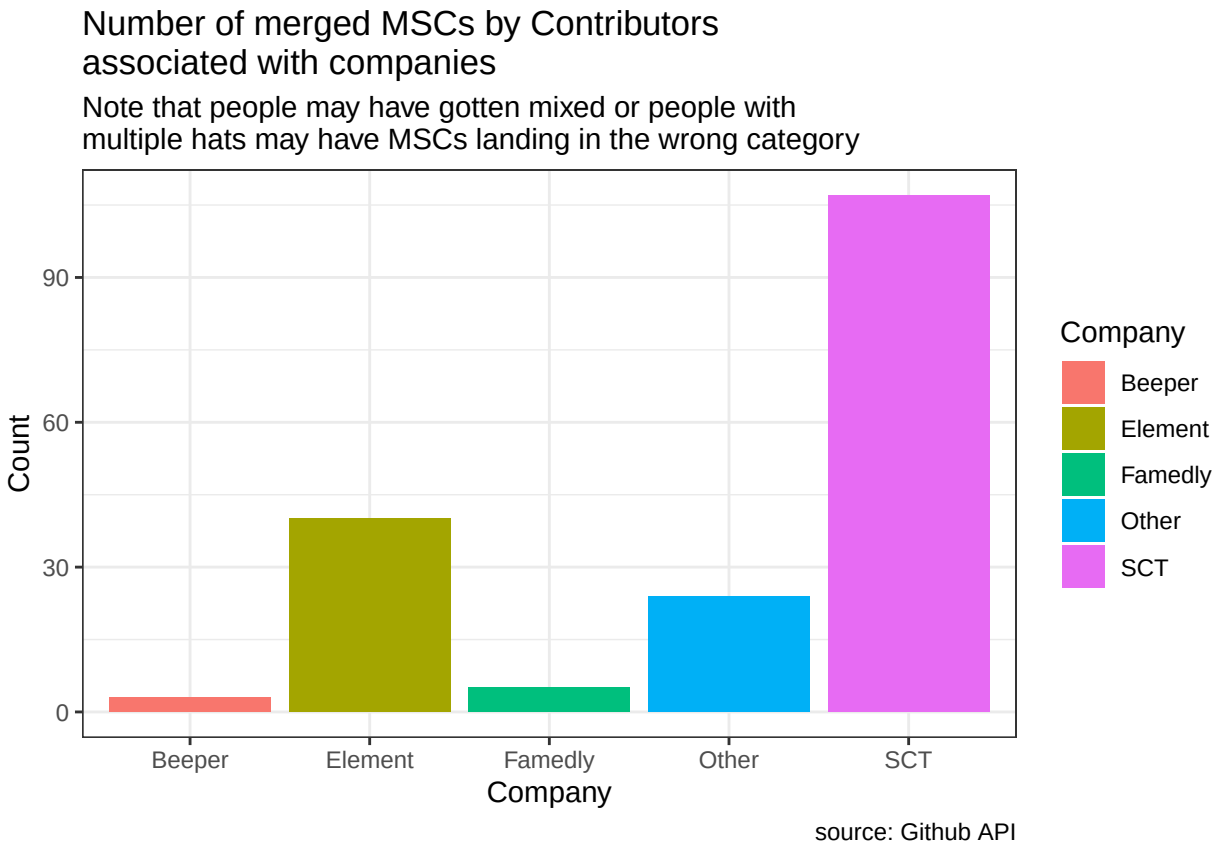
```r
data <- data.frame(column_names = column_names, Element = mscs_element, Beeper = mscs_beeper, Famedly =
data2 <- data.frame(t(data[-1]))
colnames(data2) <- data[, 1]
data <- data2
data <- cbind(Company = rownames(data), data)
rownames(data) <- 1:nrow(data)
rownames(data) <- NULL
rm(data2)


# Basic piechart
ggplot(data, aes(x = Company, y = Count, fill = Company)) +
  geom_bar(stat = "identity") +
  labs(
    title = str_wrap("Number of MSCs by Contributors associated with companies", 40),
    subtitle = str_wrap(
      "Note that people may have gotten mixed or people with multiple hats may have MSCs landing in the
      60
    ),
    caption = "source: Github API"
  )
```



Merged MSCs by Company

Note that this does not adjust for private vs company MSCs.

```
# Filter for only merged MSCs
merged_mscs <- issues_gql_all |>
  filter(!is.na(labels) && is.element("proposal", labels$name) && (is.element("disposition-merge", label

# Filter MSCs by company
merged_element <- merged_mscs |>
  filter(!is.null(author) && is.element(author, element_employee)) |>
  nrow()

merged_sct <- merged_mscs |>
  filter(!is.null(author) && is.element(author, sct_employee)) |>
  nrow()

merged_famedly <- merged_mscs |>
  filter(!is.null(author) && is.element(author, famedly_employee)) |>
  nrow()

merged_beeper <- merged_mscs |>
  filter(!is.null(author) && is.element(author, beeper_employee)) |>
  nrow()

merged_other <- nrow(merged_mscs) - merged_element - merged_beeper - merged_famedly - merged_sct

# Display Data
column_names <- c("Count")
# Display Data
data <- data.frame(column_names = column_names, Element = merged_element, Beeper = merged_beeper, Famedl
data2 <- data.frame(t(data[-1]))
colnames(data2) <- data[, 1]
data <- data2
data <- cbind(Company = rownames(data), data)
rownames(data) <- 1:nrow(data)
rownames(data) <- NULL
rm(data2)


# Basic piechart
ggplot(data, aes(x = Company, y = Count, fill = Company)) +
  geom_bar(stat = "identity") +
  labs(
    title = str_wrap("Number of merged MSCs by Contributors associated with companies", 40),
    subtitle = str_wrap(
      "Note that people may have gotten mixed or people with multiple hats may have MSCs landing in the
      60
    ),
    caption = "source: Github API"
  )
```


Number of merged MSCs by Contributors associated with companies
Note that people may have gotten mixed or people with multiple hats may have MSCs landing in the wrong category
Count
90
60
30
0
Beeper
Element
Famedly
Other
SCT
Company
Company
Beeper
Element
Famedly
Other
SCT
source: Github API