# Political Factions

Subtitle

## Contents

## 1 Introduction

In this short analysis, we use the open data from the Chamber of the Deputies of the Italian Parliament to investigate the legislative co-sponsorship of the members of the Chamber of the Deputies. The analysis would cover the legislative process that ensued under Conte I and Conte II cabinets within the legislature XVIII of Italy.

The analysis aims to understand the presence of political factions underlying the legislative process in the Lower chamber of the Italian Parliament by employing social network analysis techniques. Moreover, we will analyse the cooperative behaviour of the Parliament parties computing the *intra-opposition party bill differentiation*, as shown by De Giorgi & Dias (2018).

Lastly, this project also strives to create a fully reproducible workflow documented, commented and hosted in a public Github repository.

## 2 Methods

The analysis is based on the research produced by De Giorgi and Dias; the study analysed the network created by the co-sponsorship of legislative acts during X and Y cabinets, in office, respectively, from 1900 to 1910 and 2016 to 2000.

The study does not provide the code used to produce the analysis nor define the software used. For these reasons, the analysis has been reproduced *ex nihilo* using R statistical computing language and its packages, combined with the provided documentation.

The data has been collected using `dati.camera.it` relying on a Virtuoso Endpoint via SPARQL language. The queries were prompted through `SPARQL` packages. The Social Network Analysis was held using `igraph` package. Data processing and preparation was carried out with a selection of packages from the Tidyverse collections. As a general rule, the code was written using the Tidyverse verbs and syntax.

## 3 Technical overview

### 3.1 Package loading

```
library(here)
library(SPARQL)
library(tidyverse)
library(vroom)
library(stringr)
library(lubridate)
library(ggplot2)
library(igraph)
```

### 3.2 Data collection

The data retrieval consist of two main steps: firstly, we must declare the endpoint to scrape:

```
endpoint <- "http://dati.camera.it/sparql"
```

Since the SNA needs a list of vertices and a list of edges, we scraped the information of the deputies who took office during the selected cabinets and all the bills proposed. SPARQL queries use semantic triples, which are built as a set of three entities `subject-predicate-object`. To makes things more manageable, we firstly declare a query to get the deputies information. To restrict the result only to the 18th legislature, we used the triple `?atto ocd:rif_leg <http://dati.camera.it/ocd/legislatura.rdf/repubblica_18>`.

```
bio <- "
SELECT DISTINCT (CONCAT(?cognome,\" \",?nome) AS ?name) , ?partito, ?s_office, ?e_office
WHERE { ?persona a ocd:deputato;
        foaf:firstName ?nome;
        foaf:surname ?cognome .
          ?persona ocd:rif_mandatoCamera ?mandato .

        #adesione a partito
        ?mandato ocd:rif_deputato ?deputato .
        ?deputato ocd:aderisce ?l .
        ?l rdfs:label ?partito .
    OPTIONAL{?l ocd:startDate ?s_office.}
    OPTIONAL{?l ocd:endDate ?e_office.}

        #restict to 18esima legislatura
        ?mandato ocd:rif_leg <http://dati.camera.it/ocd/legislatura.rdf/repubblica_18> .}
ORDER BY ?name"
```

The query retrieves the deputy's name, the party, and the office's dates. The dates are pivotal since a deputy could change party during the mandate; these changes must be considered when building the social network edges.

Then we could query the database:

```
SPARQL(endpoint, bio)
```

Subsequently, we retrieved the bills. The construction of this query encountered a significant problem: the endpoint could not provide more than 10000 results at once, and we had to use a `subquery-offset` method to bypass this limitation. To restrict the result only to the Conte I cabinet, we used the triple `?atto ocd:rif_governo <http://dati.camera.it/ocd/governo.rdf/g142>`.

```
query_main <- "
SELECT DISTINCT ?num ?date(CONCAT(?primo_cognome, \" \",?primo_nome) AS ?signatory)
(CONCAT(?altro_cognome,\" \",?altro_nome) AS ?joint_signatory)
WHERE {
 {
  SELECT ?num ?date ?primo_nome ?primo_cognome ?altro_nome ?altro_cognome
  WHERE {
  ?atto a ocd:atto;
          dc:identifier ?num;
          dc:date ?date;
          ocd:rif_leg <http://dati.camera.it/ocd/legislatura.rdf/repubblica_18> ;
          ocd:rif_governo <http://dati.camera.it/ocd/governo.rdf/g142> .

  ?atto ocd:primo_firmatario ?primo .
  ?primo  a ocd:deputato;
            foaf:firstName ?primo_nome;
            foaf:surname ?primo_cognome .

  ?atto ocd:altro_firmatario ?altro .
  ?altro  a ocd:deputato;
            foaf:firstName ?altro_nome;
            foaf:surname ?altro_cognome .
    }
    GROUP BY ?atto
    ORDER BY ?num ?primo_cognome ?altro_cognome
}
  }

LIMIT 10000
OFFSET"
```

And we defined the offset limits:

```
query_offset <- c("0", "5000", "10000", "15000",
                  "20000", "25000", "30000", "35000")
```

Lastly, we build a for loop to make consecutive calls to the database; for each call, a different offset limit is concatenated to the query. This method permits retrieval of more than 10000 observations:

```
for (i in 1:length(query_offset)) {
  law <- str_c(query_main,
               query_offset[i],
               sep = " ")
  result_law <- SPARQL(endpoint, law)
  df_law <- rbind(df_law, result_law$results)
  Sys.sleep(2)
}
```

Since scraping large sets of data could be burdensome for the host, we set the offset to half of the limits (blocks of 5000 observations each), and after every call, the loop waits 2 seconds.

To retrieve the bills for Conte II cabinet, we used the same query structure, changing the "filter" triple to '`?atto ocd:rif_governo <http://dati.camera.it/ocd/governo.rdf/g162` [from `g142` to `g162`].

Each result was exported into a CSV file after the retrieval:

```r
write.csv(result1, here::here("data/deputies.csv"))
write.csv(result2, here::here("data/conte_i.csv"))
write.csv(result3, here::here("data/conte_ii.csv"))
```

The `here` package provides a relative path to the top-level directory of the project, simplifying the referencing of the data regardless of the OS or the absolute path of the directories.

### 3.3 Data preparation

## 4 Discussion

## References

De Giorgi, E., & Dias, A. (2018). Standing apart together? Common traits of (new) challenger parties in the Italian parliament. *Italian Political Science*, *13*(2).