Problem set 1 (CSS)

Solutions

1 In your own words, briefly describe what a social network is and why sociologists are interested in looking at those. **(2 points)**

- A set of acors and the connections between those. Social scientists are interested in those because the placement in a network has different implications for social phenomena, like homophily or social capital.

---

2. **2** Examine the two networks shown below.

(a) Undirected network

(b) Directed Network

Figure 1: Networkgraphs

2.1 Formally describe the networks. **(3 points)**

2.2 Provide one real-world example for each type of network shown in the figures. **(2 points)**

2.3 Based on the directed network (*figure b*), write down the corresponding adjacency matrix. **(3 points)**

```r
M <- matrix(c(0,0,0,0,0,0,0,0,0,
       0,0,0,0,0,0,0,0,1,
       1,0,0,0,0,1,0,0,0,
       0,0,0,0,0,0,0,1,0,
       0,0,1,0,0,1,0,0,0,
       0,0,1,0,0,0,0,1,1,
       1,1,1,0,0,0,0,1,1,
       0,1,0,0,0,1,0,0,0,
       1,1,0,0,0,1,0,0,0),
       nrow = 9)

p <- graph_from_adjacency_matrix(
  M,
  mode = "directed"
)
```

---

3 A city partnership (also known as a twin city or sister city relationship) is a formal agreement between two cities (most of the time from two different countries) to promote cultural and economic collaboration. The city of Leipzig has 15 such twin cities. On Moodle, you will find an R data file named `twin_cities.rds`, which contains a list of Leipzig's partner cities and their own partner cities. You can load the file into R using the following command:

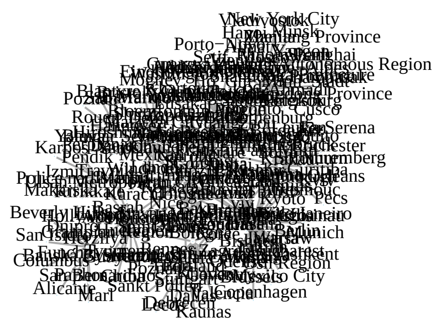
`twin_cities <- readRDS("your/path/twin_cities.rds")`

3.1 Using this data, create an igraph object where each node represents a city. Add an edge between two cities if they are partnered and include the country of each city as a vertex attribute. **(5 points)**

```r
twin_cities <- readRDS("Data/twin_cities.rds")

# 1. Create a data frame of edges
edges <- data.frame(from=character(), to=character(), stringsAsFactors = FALSE)

# 2. Create an empty vector for the countries
countries <- vector("list", length = length(twin_cities))

# 3. Fill the edges and countries list
for (city in names(twin_cities)) {
  for (twin in twin_cities[[city]]$cities) {
    edge <- sort(c(city, twin))  # Sort to avoid duplicate directions
```

```r
    edges <- rbind(edges, data.frame(from=edge[1], to=edge[2], stringsAsFactors = FALSE))
  }
}

# 4. Remove duplicate edges
edges <- unique(edges)

#unique(edges$from)

# 5. Create a list of countries for each city using their indices
city_countries <- unlist(lapply(names(twin_cities), function(city) twin_cities[[city]]$countr


# 6. Create the graph
g <- graph_from_data_frame(edges, directed = FALSE)

# 7. Add countries as vertex attributes
V(g)$country <- city_countries[match(V(g)$name, names(twin_cities))]



plot(g,
     vertex.size = 2,
     vertex.label.cex = 0.6,
     vertex.label.color = "black",
     vertex.label.dist = 1,
     vertex.color = "skyblue",
     vertex.frame.color = "white",
     layout = layout_with_lgl(g),
     main = "City Network with Countries"
)
```

City Network with Countries



3.2 Categorize the form of data collection. **(1 point)**

3.3 Calculate the density of the network. Then explain in one or two sentences what this value tells you about the structure of the network. **(2 points)**

```r
edge_density(g)
```

```
[1] 0.01129418
```

3.4 Determine the diameter of the network. What does this diameter represent in the context of the city partnership network? **(1 point)**

```r
diameter(g)
```

```
[1] 4
```

3.5 Find the shortest path between the cities Budapest and Wuhan. List all cities that lie along this path. **(1 point)**

```r
# Find the shortest path between Hamburg and Tel Aviv
shortest_path <- shortest_paths(g, from = "Budapest", to = "Wuhan", output = "both")

# Print the shortest path
print(shortest_path)
```

```
$vpath
$vpath[[1]]
+ 5/213 vertices, named, from 3acb01b:
[1] Budapest          Frankfurt_am_Main Kraków            Kyjiw
[5] Wuhan


$epath
$epath[[1]]
+ 4/255 edges from 3acb01b (vertex names):
[1] Budapest         --Frankfurt_am_Main Frankfurt_am_Main--Kraków
[3] Kraków           --Kyjiw             Kyjiw            --Wuhan


$predecessors
NULL

$inbound_edges
NULL
```

```r
# Extract the vertices in the shortest path
path_vertices <- shortest_path$vpath[[1]]

# Print the cities (vertices) in the shortest path
print(V(g)[path_vertices]$name)
```

```
[1] "Budapest"          "Frankfurt_am_Main" "Kraków"
[4] "Kyjiw"             "Wuhan"
```

3.6 Discribe the distribution of shortest paths in the network. Then interpret what this distribution tells you about the overall connectivity of the cities. **(*2 points)**