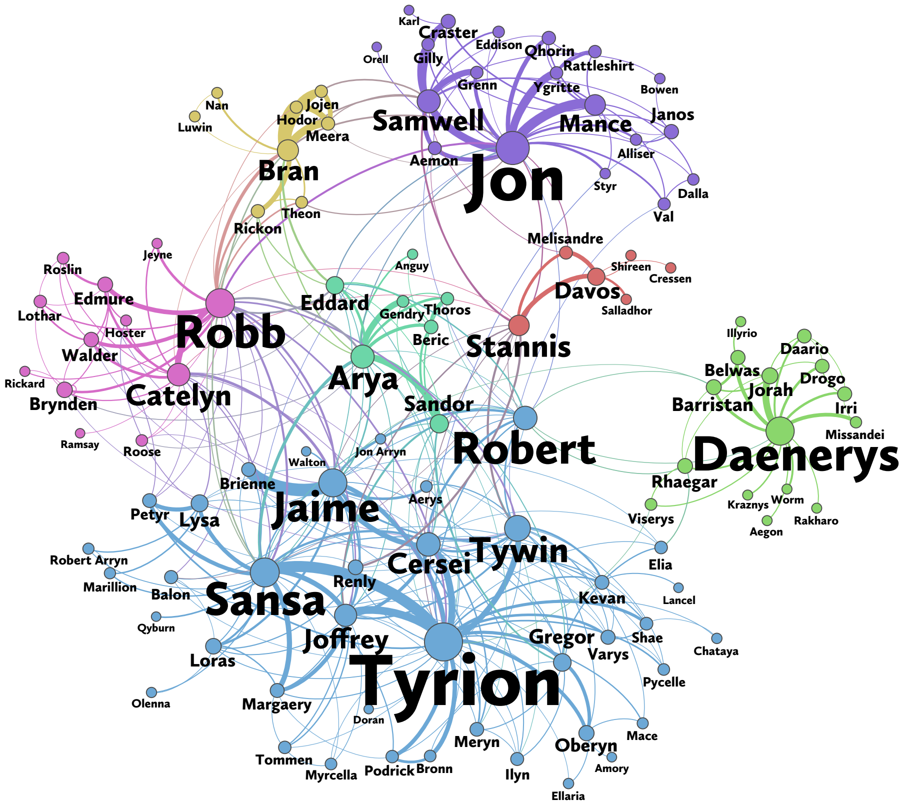
PNAWS 2019
Exercises 1.2

Network Visualization

In these exercises, we will analyse a weighted social network of book series *A Song of Ice and Fire*, on which the popular TV show *Game of Thrones* is based. This network was recently published in Math Horizons Magazine (Beveridge & Shan, 2016), and is based on the third book, A Storm of Swords, on which the third and fourth season of the TV series are based. More information, including the data, is available online. The network published is the following:



Exercise 1 Look through the paper to get an idea of what this network represents. What do the nodes and edges represent?

The data as published online can be loaded in R as follows:

```
Data <- read.csv("stormofswords.csv")
```

Exercise 2 Look at the data in RStudio using the 'View' function. This matrix encodes a network. Can you figure out how? What do the rows stand for and what do the columns stand for?

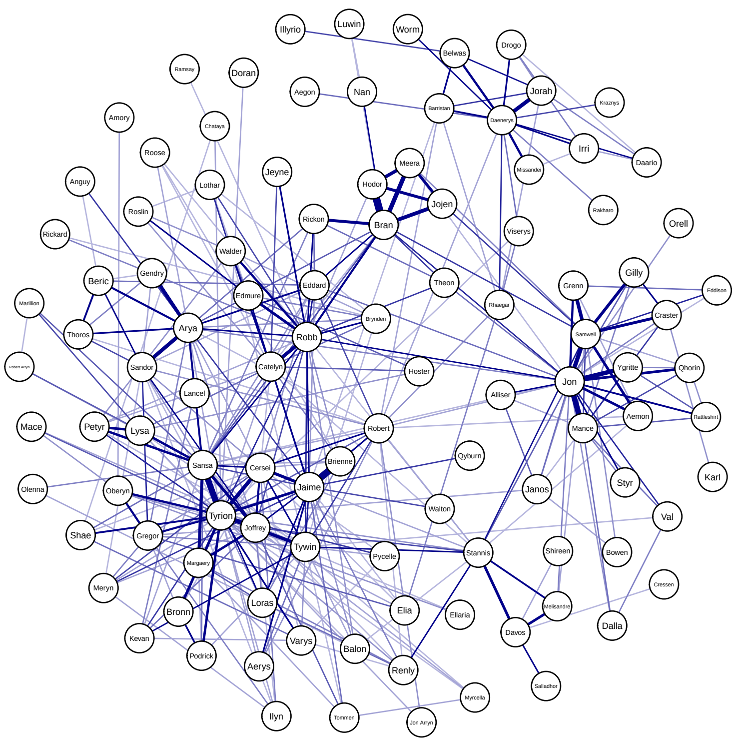
This structure is known as an *edgelist* encoding a network, which can also be used as input for qgraph:

```
library("qgraph")
qgraph(Data, directed = FALSE)
```

Exercise 3 When plotting an undirected graph using an adjacency or weights matrix as input I normally do not have to set the `directed` argument? Now that I use an edgelist, however, I do. Why?

The plotted network is plotted using a circular layout. This circular layout, however, is hard to interpret and read in such networks with many nodes. In addition, I like to plot edges with a different color than green when they represent values that can only be positive. Finally, the nodes are very large and we might want to make them smaller:

```
## Registered S3 methods overwritten by 'huge':
##   method    from
##   print.sim BDgraph
##   plot.sim  BDgraph
```



Exercise 4 Recreate the plot above, changing the layout to a spring layout, the edge color to '"darkblue"' and the node size to 3. Look at the qgraph help page to figure out the commands needed (`?qgraph`). Note: your computer might generate a different spring layout (nodes placed on different locations). ■

qgraph uses three arguments that need to be known to interpret a network: `minimum`, `cut`, and `maximum`. These are set automatically, and can be shown using the argument `details = TRUE`.

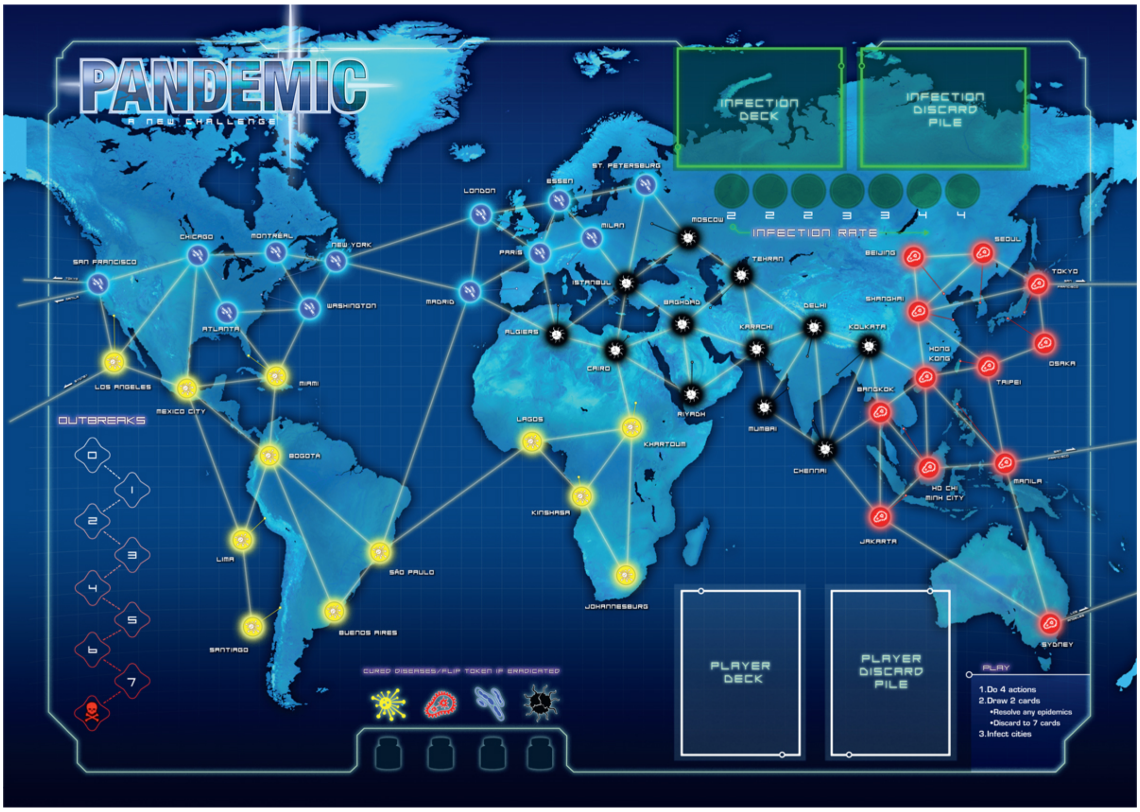
Exercise 5 What values did qgraph set to `cut` and `maximum`? Note: `minimum` is always set to 0 by default and not shown with `details = TRUE` unless it differs from 0. ■

The `minimum` argument can be used to *hide* edges with an absolute (negative edges are treated as positive) weight under some value. Note that these edges are only visually hidden, not removed in further analyses (which can be done using the `threshold` argument). This argument is useful when plotting dense graphs (e.g., correlation networks) but *not* recommended in the networks estimated in this course.

Exercise 6 Set the `minimum` argument to 1, 5 and 10 while using a spring layout. How does the network change? Do the same using the `threshold` argument. Can you explain why the layout remains the same using `minimum` but changes using `threshold`? ■

Network Inference

Sometimes, some of the PNASS teachers get together to play the board game Pandemic. In this game, the goal is to cooperate to save the world from four vicious viruses. The board itself is a network:



A blue virus will emerge in the blue cities (nodes), a yellow virus in the yellow cities, a black virus in the black cities and a red virus in the red cities. Once there is an outbreak of a virus in a city, it may spread to connected city via the edges. Fortunately, we have some tools to stop the viruses from spreading. For example, we can place a city under quarantine, effectively removing the node from the network. Unfortunatly, we frequently lose and are unable to save the world. Therefore, we need your help!

Exercise 7 Only by looking at the above picture, can you derive:

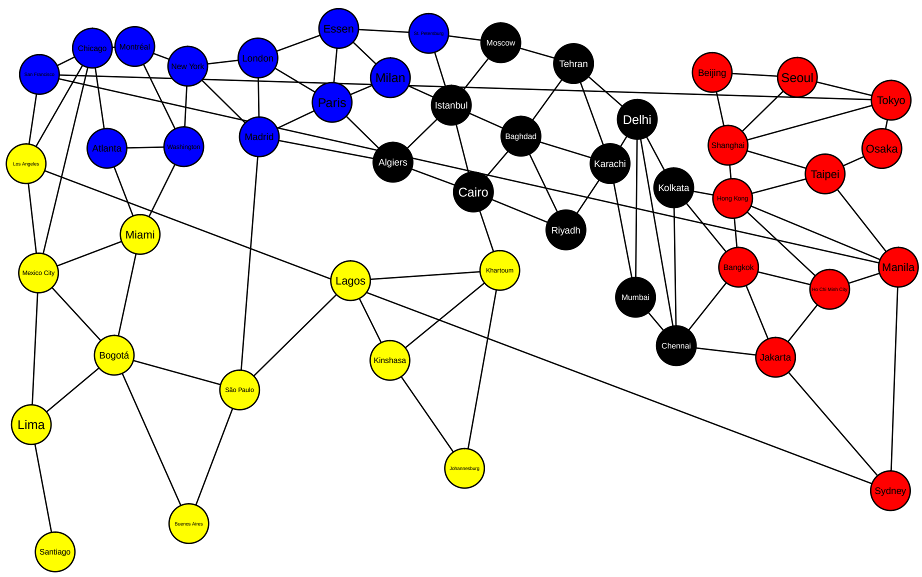
- The degree-centrality of Paris?
- The shortest path length between Milan and New York, and how many paths there are of this length?
- Of all shortest paths between all pairs of nodes, how many go through Santiago?

Luckily, we are not the only ones who overanalyze boardgames and the hard work of encoding the network structure was already done online.[1] We can read the network into R using the following codes:

```r
# Load network:
library("igraph")
Graph <- read_graph("http://files.indicatrix.org/pandemic.graphml",
          format = "graphml")

# Extract edgelist:
Edgelist <- get.adjacency(Graph)

# Labels:
Labels <- V(Graph)$label
```

[1] https://indicatrix.org/overanalyzing-board-games-network-analysis-and-pandemic-482b2018469

```r
# Layout:
Layout <- cbind(x = V(Graph)$x,
                y = V(Graph)$y)

# Color:
Color <- c(
  rep("blue",12),
  rep("black",12),
  rep("red",12),
  rep("yellow",12)
)

# Plot in qgraph:
library("qgraph")
qgraph(Edgelist, labels = Labels, color = Color,
       layout= Layout, directed = FALSE,
       vsize = 4, esize = 1, edge.color = "black")
```



I used the Layout argument to manually assign a layout that corresponds to the board. However, looking at the board I would have liked to place the node "Jakarta" a bit lower.

Exercise 8 Make a change to the `Layout` argument to place the node Jakarta a bit lower. ■

Now that we can visualize the network, we might wish to analyze it further. To do this, we first need to store the network:

```r
Graph <- qgraph(Edgelist, labels = Labels, color = Color,
       layout= Layout, directed = FALSE,
       vsize = 4, esize = 1, edge.color = "black")
```

One option is to analyze how important or *central* nodes are in the network.[2] These can be analyzed using the `centrality` function:

[2] A detailed description of how these measures can be computed can be found in Opsahl, Agneessens, and Skvoretz (2010)

```
Centrality <- centrality(Graph, all.shortest.paths = TRUE)
```

Node strength (also called degree) sums the connected edge weights to a node. The node strength of the city Miami is:

```
Miami <- which(Labels == "Miami")
Centrality$OutDegree[Miami]

## Miami
##     4
```

Closeness computes how “close” two nodes are together in the network:

```
Centrality$Closeness[Miami]

##       Miami
## 0.005291005
```

Finally, *Betweenness* computes how often one node is featured in the most efficient (shortest) paths between other node:

```
Centrality$Betweenness[Miami]

##    Miami
## 42.81905
```

Exercise 9 Compare the centrality of the cities named “Bangkok” and “Atlanta” ■

Exercise 10 List the most and least central node according to degree, closeness and betweenness. If multiple nodes are equally central, list them all. ■

I can investigate the degree of all red virus cities as follows:

```
Red <- which(Color == "red")
Centrality$OutDegree[Red]

##           Beijing             Seoul             Tokyo          Shanghai
##                 2                 3                 4                 5
##         Hong Kong           Taipei             Osaka           Bangkok
##                 6                 4                 2                 5
## Ho Chi Minh City            Manila           Jakarta            Sydney
##                 4                 5                 4                 3
```

Exercise 11 Compute the average degree, closeness and betweenness per virus (color). Which virus do you think is the most dangerous? ■

Challenge questions

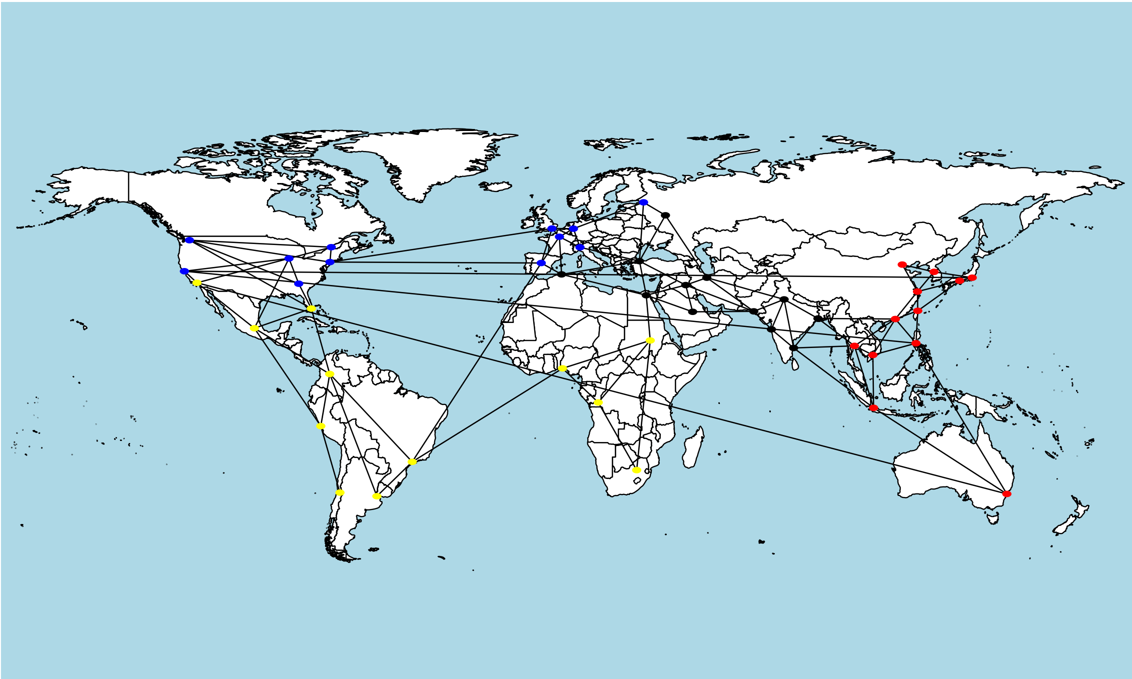
Challenge question 1

If we can quarantine a city, then it is effectively removed from the network. As a result, removing one node affects the centrality of other nodes. Write a script that automatically removes nodes from the network based on the centrality. You can choose any strategy. For example, you may choose to select the node with the

highest degree, and if multiple nodes have the same degree select one at random or select of those nodes the node with the highest closeness. Automatically recompute the network and centrality indices and repeat the script to remove another node until you have removed 5 nodes from the network. Which nodes did you remove?
Tip: it may be useful to use an adjacency matrix instead of edgelist as input. You can obtain an adjacency matrix with `getWmat(Graph)`

Challenge question 2

When plotting a geographical network, you may want to plot the network on the actual map. Overlay a world-map with the pandemic network you used in this assignment. You may use any tools you wish and make the plot as fancy as you want. My own version is:



Note and tips: you need the actual geographical locations of the cities as layout, which you do not have yet (the layout used is approximate but not exact). I obtained these via R. Furthermore, there are ways to plot a world map in R (Google!). You can overlay an existing plot with a qgraph network using qgraph arguments`plot` and `rescale`.

References

Beveridge, A., & Shan, J. (2016). Network of thrones. *Math Horizons*, *23*(4), 18–22.
Opsahl, T., Agneessens, F., & Skvoretz, J. (2010). Node centrality in weighted networks: Generalizing degree and shortest paths. *Social networks*, *32*(3), 245–251.