STAT 332: Network Analytics Homework 3

Instructor: Joseph Reid

2026-01-19

Instructions:

Please complete this homework assignment by the due data posted on canvas. Please submit your work in an html file with solutions and code presented completely.

Homework 2: Networks and visualization with igraph

This homework is focused on plotting with ggraph and visNetwork but will also be used to explore the tidygraph framework. Tidygraph is a library which provides a subclass of igraph and enables much smoother manipulation of graph objects using the tidyverse methodologies.

First, make sure to install the networkdata package. This package isn't located in CRAN so it takes a little bit of work using remotes library to get. If you don't have remotes, install that first. Then, install network data using these instructions: https://www.rdocumentation.org/packages/networkdata/versions/0.1.6

1.) Getting the Data

```
library(networkdata)
library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```
?wta
```

```
## starting httpd help server ...

##  done
```

We are going to use the wta dataset for this. Specifically, we are going to use the data from 2007. Extracting this data isn't difficult but it is a little weird. The wta dataset is a list of igraphs, so you would think all that's necessary to get the first item in the list is to use wta[1]. Try a summary of this.

```
summary(wta[40])
```

```
##      Length Class  Mode
## [1,] 255    igraph list
```

Interestingly, you didn't get the igraph! You got the list item but not what's in it. To get that, you actually need to use a double bracket: wta[[1]].

```
class(wta[[40]])
```

```
## [1] "igraph"
```

So, this is a directed, named, weighted dataset with 255 players and 1771 edges where the edge weights denote the number of times the "from" player lost to the "to" player on a specific surface type. Let's store this as g.

```
g <- wta[[40]]
```

Now, getting a look at the actual values of the matrix of node attributes or edge attributes is actually not obvious; however, if we convert this using a as_tbl_graph with tidygraph, we can make this much easier.

```
library(tidygraph)
```

```
##
## Attaching package: 'tidygraph'

## The following object is masked from 'package:igraph':
##
##     groups

## The following object is masked from 'package:stats':
##
##     filter
```

```
g2 <- as_tbl_graph(g)
```

Now, these objects are interesting because we can activate either the nodes or the edges (whatever we want to work with.) When we work with one set (for example, by applying filtering), the other set is also affected. We have to be careful though, because edge filtering won't necessarily remove all corresponding nodes, but removing a node will absolutely remove all edges attached to that node. We should note that there are 55 total countries represented here. This is going to be a bit much to draw, so we're going to limit our dataset.

```
length(unique(V(g2)$country))
```

```
## [1] 55
```

In order to perform operations on this, we need to activate either nodes or edges and then apply our filter operations using "filter". This works exactly the same as it does in the tidyverse methodology and can be found easily online.

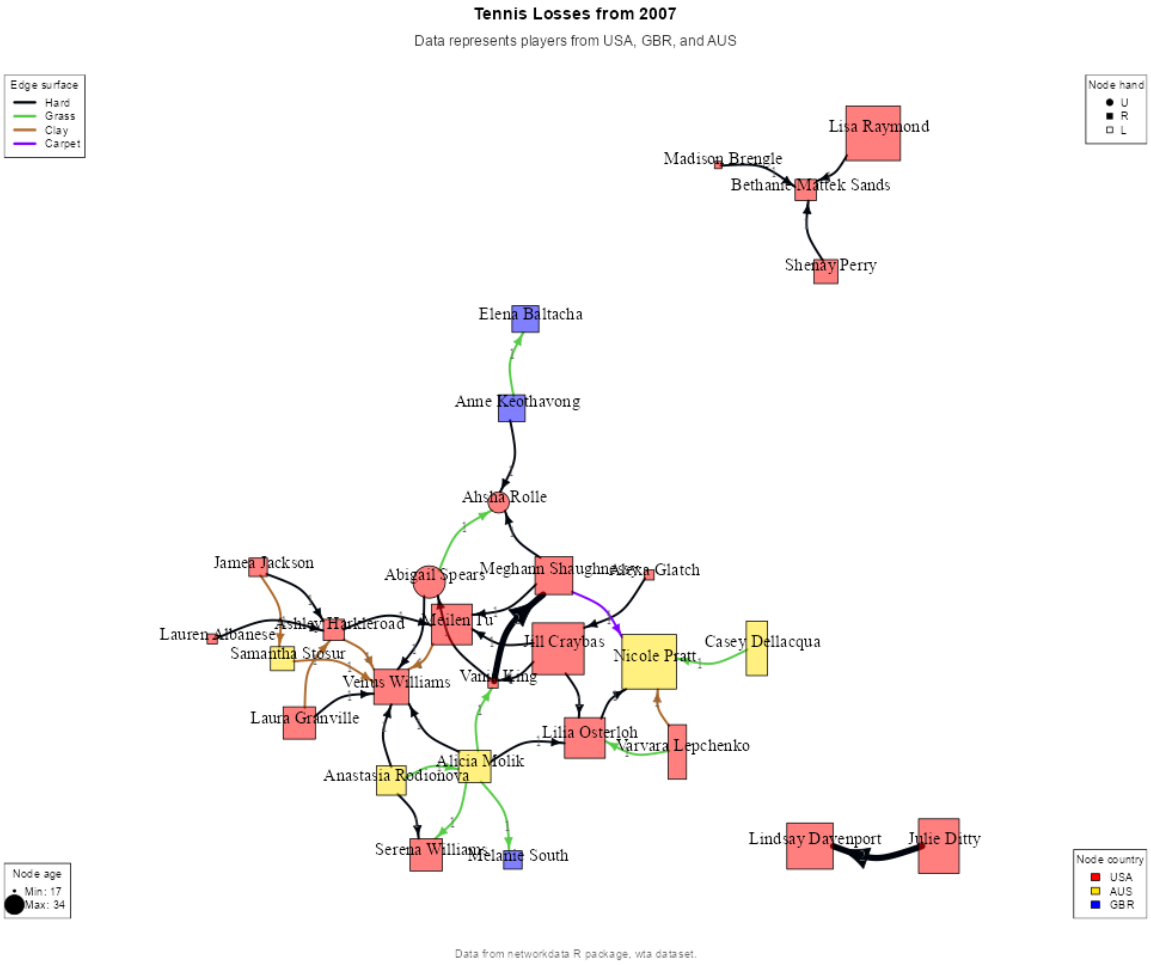
```
g3 <- g2 %>%
  activate("nodes") %>%
  filter(country %in% c("USA", "GBR", "AUS")) %>%
  filter(!node_is_isolated())
```

Saving your data at this point is a good idea. You can do this with write_graph() as follows.

```
write_graph(g3, format = "gml", file = "Homework_3_filtered_data.gml")
```

Now, it's time to do our work::

2.) Graph with ggraph



Your job is to produce the code/graph which reproduces or improves on this graph using the ggraph and visNetwork packages. Good luck!

3.) Graph with visNetwork