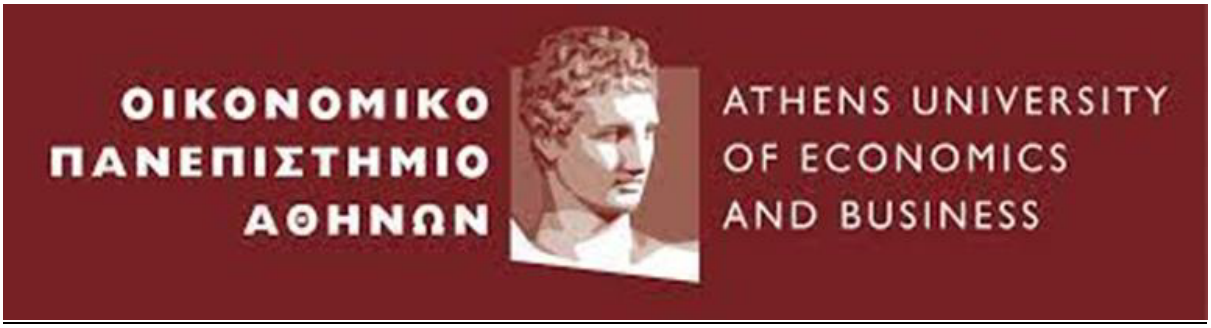


ATHENS UNIVERSITY OF ECONOMICS & BUSINESS
DEPARTMENT OF MANAGEMENT, SCIENCE & TECHNOLOGY
MSc BUSINESS ANALYTICS

Project 1
Network Analysis and Visualization with R and igraph

Course: Social Network Analysis – Spring 2023

Full Name: ATHANASIOS ALEXANDRIS
Register Number: p2822202

Table of Contents

Abstract

Using R and iGraph, we analyzed the interactions between characters in George R. R. Martin's series "A Song of Ice and Fire". Our first step was to create an undirected and weighted graph. We then calculated basic network properties such as the number of vertices, edges, diameter, and triangles in the graph. Additionally, we identified the Top 10 characters based on their degree, weighted degree, and local clustering coefficient. To better understand the network, we plotted both the entire graph and a subgraph consisting of vertices with 10 or more connections, and compared their edge density. We also used betweenness centrality and closeness centrality metrics to identify the Top 15 characters, and compared Jon Snow's rankings in these metrics. Finally, we used the PageRank algorithm to rank the characters and visualize the results in a graph.

1. Task 1

In the first task, we downloaded the csv file that contained the list of edges of the network. We used the columns Source, Target, and Weight as input data frame in the graph_from_data_frame() function, and we created an undirected weighted igraph graph (ig).

2. Task 2

In the second task, used the proper code to demonstrate the basic properties of our network. Our network is consisted of 796 vertices (vcount(ig)) and 2823 edges (ecount(ig)), and its diameter is 53 (diameter(ig)).

Next we calculated the number of triangles in our graph, which is 16.965 (length(triangles(ig))). However, as our graph is undirected, triangles with just different order, have been counted thrice but actually they are exactly the same. For instance the triangles [Tyrion-Lannister , Sansa-Stark, Jon-Snow] is the same as the triangles [Sansa-Stark, Jon-Snow, Tyrion-Lannister] and [Jon-Snow ,Tyrion-Lannister , Sansa-Stark]. Hence we divided the total number of triangles in the graph by 3, to find the number of unique triangles, which is 5655.

Next we used the function sum(E(ig)$weight > 15), to find all edges that have weight greater than 15. The outcome was 478 edges.

Moreover, we had to find the top-10 characters of the network (vertices), based on their degree, weighted degree and local clustering coefficient.

For the first top-10, we used the degree() function, to get the degree score for each character, then we sorted it by descending order and kept the first 10 observations. The degree score of each vertex the number of edges that are incident to that vertex. In our undirected graph, the degree of a vertex is just the number of edges that connect to it.

For the first top-10, we used the strength() function, to get the weighted degree score for each character, then we sorted it by descending order and kept the first 10 observations. The weighted degree of a vertex in a graph is a variant of the degree that takes into account the weights of the edges incident to the vertex.

The outcomes are presented below:

Degree	
Tyrion-Lannister	122
Jon-Snow	114
Jaime-Lannister	101
Cersei-Lannister	97
Stannis-Baratheon	89
Arya-Stark	84
Catelyn-Stark	75
Sansa-Stark	75
Eddard-Stark	74
Robb-Stark	74

Wighted degree	
Tyrion-Lannister	2873
Jon-Snow	2757
Cersei-Lannister	2232
Joffrey-Baratheon	1762
Eddard-Stark	1649
Daenerys-Targaryen	1608
Jaime-Lannister	1569
Sansa-Stark	1547
Bran-Stark	1508
Robert-Baratheon	1488

As we observe the most important character based on both degree and weighted degree is Tyrion Lannister. In the second place to both categories, is Jon Snow. Additionally, it is important to mention that the characters Jamie Lannister, Cersei Lannister, Eddard Stark and Sansa Stark are included in both top-10 lists. All these characters that we mentioned, have many and strong connections in the network, that makes them the most important ones, in terms of these metrics.

The local clustering coefficient of a vertex in a graph is a measure of how "close-knit" the neighborhood of that vertex is. It is defined as the ratio of the number of edges that actually exist between the neighbors of the vertex to the total number of possible edges between them. The local clustering coefficient can be used to measure how much a vertex contributes to the clustering of a graph, and is often used to identify nodes that are likely to be part of cliques or other types of tightly-knit communities. We observed that the highest score is 1.

We used the transitivity() function to get the score for each character, and then we sorted it by descending order. We observed that the highest score is 1, and that 178 characters have this score, so we could not end up to a Top-10. Below is presented the list of the 178 characters mentioned above.

Aegon-Frey-(son-of-Stevron)	Frenya	Mellei	Morra
Albett	Fulk	Mezzara	Kraznys-mo-Nakloz
Alerie-Hightower	Garth-(Wolfs-Den)	Moelle	Brusco
Allar-Deem	Garth-Greyfeather	Morrec	Talea

Alys-Karstark	Gendel	Murenmure	Ternesio-Terys
Alysane-Mormont	Genna-Lannister	Mycah	Vayon-Poole
Amabel	Gorne	Olyvar-Frey	Waif
Arron	Gorold-Goodbrother	Orell	Tristifer-Botley
Baelor-Blacktyde	Grunt	Osfryd-Kettleblack	Porridge
Baelor-I-Targaryen	Guncer-Sunglass	Oswell-Kettleblack	Steelskin
Balman-Byrch	Guyard-Morrigen	Palla	Oznak-zo-Pahl
Bannen	Gyles-Grafton	Patchface	Mariya-Darry
Bedwyck	Hairy-Hal	Perwyn-Frey	Wick-Whittlestick
Ben-Bones	Hallyne	Prendahl-na-Ghezn	Howland-Reed
Benethon-Scales	Hero	Preston-Greenfield	Reek
Beth-Cassel	Heward	Pyat-Pree	Stiv
Big-Boil	High-Septon-(fat_one)	Qarl-the-Maid	Illifer
Brella	High-Septon-(Tyrions)	Quellon-Greyjoy	Narbert
Brus-Buckler	Hop-Robin	Quhuru-Mo	Willow-Heddle
Brynden-Rivers	Humfrey-Hewett	Ralf-Stonehouse	Robin-Ryger
Butterbumps	Husband	Raymun-Darry	Utherydes-Wayn
Cedra	Jarman-Buckwell	Reznak-mo-Reznak	Senelle
Chayle	Jommo	Rhaella-Targaryen	Vylarr
Chella	Jon-Umber-(Smalljon)	Rhaenys-Targaryen	Ghael
Chiggen	Jonos-Bracken	Richard-Horpe	Sallor
Clayton-Suggs	Jonothor-Darry	Robin-Flint	Symon-Stripeback
Clement-Piper	Joramun	Rodrik-Sparr	Mag-Mar-Tun-Doh-Weg
Cley-Cerwyn	Joseth	Roger-Ryswell	Willam-Dustin
Clubfoot-Karl	Josmyn-Peckledon	Roslin-Frey	Forley-Prester
Cohollo	Jyck	Ryk	Will-(prologue)
Colen-of-Greenpools	Kasporio	Scar	Todder
Cromm	Kedry	Scolera	Rory
Dacey-Mormont	Khorane-Sathmantes	Sweets	Myles-Toyne
Daemon-Sand	Kindly-Man	Symon-Silver-Tongue	Sigfryd-Harlaw
Dalbridge	Kojja-Mo	Theomore	Ronnet-Connington
Daryn-Hornwood	Koss	Toregg	Wylis-Manderly
Daven-Lannister	Leaf	Tybero-Istarion	Urzen
Denyo-Terys	Leona-Woolfield	Walder-Frey-(son-of-Jammos)	Xhondo
Desmond	Lyonel-Corbray	Woth	Yellow-Dick
Dick-Follard	Maggy	Torrhen-Stark	Rollam-Westerling
Dirk	Marlon-Manderly	Visenya-Targaryen	Steffon-Baratheon
Drennan	Marselen	Ysilla	Walda-Frey-(daughter-of-Merrett)
Elder-Brother	Marya-Seaworth	Hayhead	Werlag
Emrick	Megga-Tyrell	Harra	
Eon-Hunter	Meldred-Merlyn	Tickler	

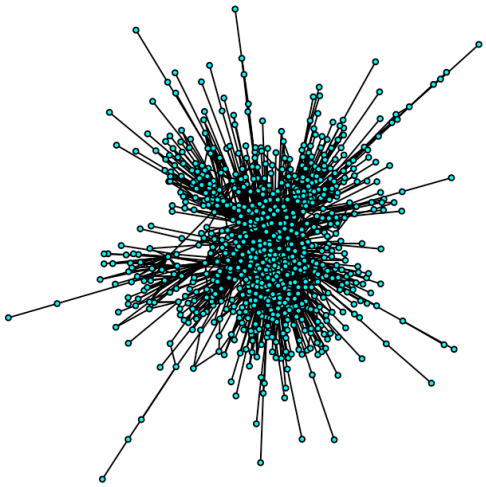
As we observe none of the characters that were included in the degree and weighted degree Top 10s are included in this list. This can be explained, as nodes with high degree and weighted degree tend to be "hubs" in the graph, meaning they have many direct connections to other nodes. However, these connections may not necessarily lead to a tightly-knit community or cluster of nodes, which is what transitivity measures. In fact, hubs may even serve as bridges between different clusters in the graph, which can actually lead to lower transitivity.

Finally we calculated the global clustering coefficient of the graph, using the transitivity() function, setting the option “global” in the “type” argument. The outcome is 0.2090367, that indicates that a small fraction of the possible triangles in the graph are actually present. Nodes in the graph are not strongly clustered together, which means that the graph is relatively sparse or that the nodes are well-connected in smaller subgroups but not in the graph as a whole.

3. Task 3

In the third task, first we had to plot the entire network. The plot of our entire network is presented below:

Figure 1. – Entire Network

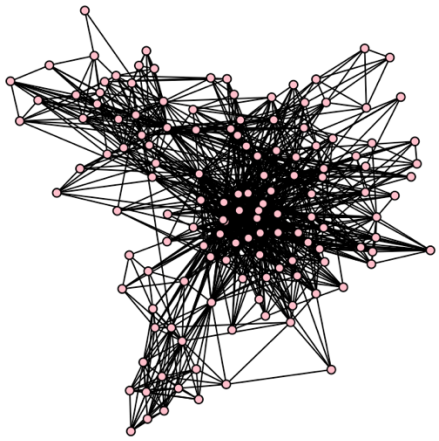


Next, we had to create a subgraph of our network(), that includes all vertices that have more or equal than 10 connections in the network, and plot the subgraph.

We created the subgraph by using the delete_vertices(sub_ig) function, setting as argument the vertices with degree below to 10.

The plot of our subgraph is presented below:

Figure 1. – Sub. Network



Then we used the edge_density() function, to calculate the edge density for the entire graph and the subgraph.
The density for the entire network is 0.0089 and for the sub graph is 0.117.

The density of a graph is defined as the ratio of the number of edges in the graph to the maximum possible number of edges for that graph. For the entire network, the density is small, which means that the network is relatively sparse. The density of the subgraph with vertices that have 10 connections or more, is significantly higher than the density of the entire network, indicating that the subgraph is more densely connected than the entire network, and that there are more tightly-knit communities or clusters of nodes in the subgraph.

1. Task 4

In the fourth task, first we calculated the top-15 nodes according to the closeness centrality, and betweenness centrality.

For the closeness centrality we used the closeness() function, to get the closeness score for each character, then we sorted it by descending order and kept the first 15 observations.

Nodes with high closeness centrality are typically more central to the network, which means that they can access other nodes more quickly and easily than nodes. These nodes are very important for the network, and their removal from the network could have significant effects on network connectivity.

For the betweenness centrality we used the closeness() function, to get the betweenness score for each character, then we sorted it by descending order and kept the first 15 observations.

Nodes with high betweenness centrality are typically located on many of the shortest paths between pairs of nodes in the network. These nodes may act as "bridges" or "hubs" between

different parts of the network, and their removal from the network could have significant effects on network connectivity.
The outcomes are presented below:

Closeness Centrality	
Jaime-Lannister	0.0001205982
Robert-Baratheon	0.0001162791
Stannis-Baratheon	0.0001146921
Theon-Greyjoy	0.0001146132
Jory-Cassel	0.0001141553
Tywin-Lannister	0.0001137656
Tyrion-Lannister	0.0001130071
Cersei-Lannister	0.0001129688
Brienne-of-Tarth	0.0001124480
Jon-Snow	0.0001118944
Joffrey-Baratheon	0.0001105094
Rodrik-Cassel	0.0001103631
Eddard-Stark	0.0001092180
Doran-Martell	0.0001088613
Robb-Stark	0.0001088495

Betweeness Centrality	
Jon-Snow	41698.94
Theon-Greyjoy	38904.51
Jaime-Lannister	36856.35
Daenerys-Targaryen	29728.50
Stannis-Baratheon	29325.18
Robert-Baratheon	29201.60
Tyrion-Lannister	28917.83
Cersei-Lannister	24409.67
Tywin-Lannister	20067.94
Robb-Stark	19870.45
Arya-Stark	19354.54
Barristan-Selmy	17769.29
Eddard-Stark	17555.36
Sansa-Stark	15913.44
Brienne-of-Tarth	15614.41

In terms of closeness centrality we observe that the most important character is Jamie Lannister and in terms of betweenness centrality the most important character is Jon Snow. Both characters are included in both Top-15 lists, along with the characters Robert Baratheon, Stannis Baratheon, Theon Greyjoy Tyrion Lannister, Cersei Lannister, Brienne of Tarth, Eddard Stark and Rob Stark.

Next we calculated the closeness centrality(0.0001118944) and betweenness centrality for John Snow (0. 41698.94), using the same function, setting the argument v (vertices) to be equal to John Snow. Then we find the ranking of John Snow for the two above measures.

John Snow is in the 10^{th} place in terms of closeness centrality and first in terms of betweenness centrality. This makes John Snow maybe the most important character in the network, since has the higher betweenness of all the other characters and one the highest closeness in the network. That indicated that he is well-connected within his immediate neighborhood and also plays a crucial role in facilitating communication between different parts of the network. This character can be consider of as a central hub in the network that connects many different nodes, while also being well-connected to its immediate neighbors.

1. Task 5

In the final task, we had to rank the characters of the network with regard to their PageRank value.
First we used the pagerank() function to calculate the page rank of each character. Then we plot the graph setting in the vertex.size argument the product of the PageRank vector of each character with 400, so that we can create a size scale in our plot, where the bigger the Pagerank of the character the bigger the node in the graph.

Our plot is presented below

Figure 1. – Sub. Network