Analyzing Narratives in Social Networks_Ex1

Analyzing The " The Trial of the Chicago 7" Movie

Part 1

a. We selected the movie The Trial of the Chicago 7.

b. **Summary**: During the democratic convention in Chicago, protests against the ongoing war in Vietnam get violent and bloody. Several months after that, after elections, the U.S Department of Justice puts the protest leaders on trial. Throughout the trial we learn more about the protests, and the organizers.

c. **The main characters:**
   a. Abbie Hoffman – the leader of the hippy protests
   b. Tom Hayden – the leader of the students (more proper) protests
   c. William Kunstler – the lawyer defending the Chicago 7
   d. Richard Schultz – the prosecutor + maybe the Judge Hoffman.

d. **Weaknesses:**
   a. Abbie Hoffman:
      his weakness is the concept of establishment. He's in general anti-government and rules and believes in being more open.
   b. Tom Hayden:
      his weakness is that he believes in the system, yet the system is the one who at first didn't let him protest, and then arrested him and put him on trial.
   c. William Kunstler:
      his weakness is the obvious prejudice that Judge Hoffman has in the trial. With that, he sometimes phrases himself in a manner not appropriate for an attorney in a courthouse.
   d. Richard Schultz:
      his weakness is he doesn't completely believe the defendants are guilty.

e. **The AB graph:**
   this graph connects between the speakers in the movie. Generally speaking, it is safe to assume that people who speak one after the other are speaking to each other and are in the same party. When adding weights, we can see who speaks more, and with this determine who the main characters are, and who they are most connected to.

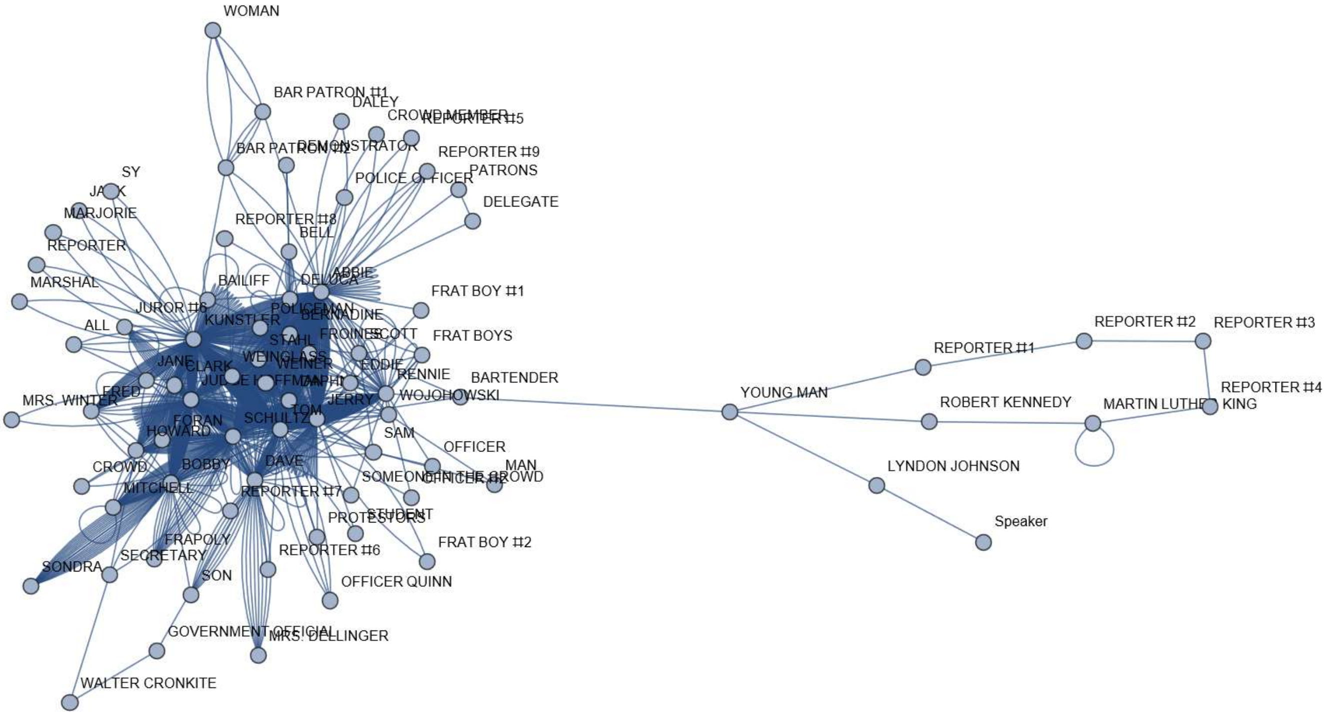
f. **Compute the AB.csv file** :
   For this part, we started with the pdf found online. We used an online tool to convert it to .txt, and then with a python script we turned into csv. This script realized that the speaker is always upper case indented with quite a bit of tabs. We ran over all lines, grabbed the speakers, saved the text brought after that, and exported to csv file. (The

python code is attached -called "*parse_script.py*").

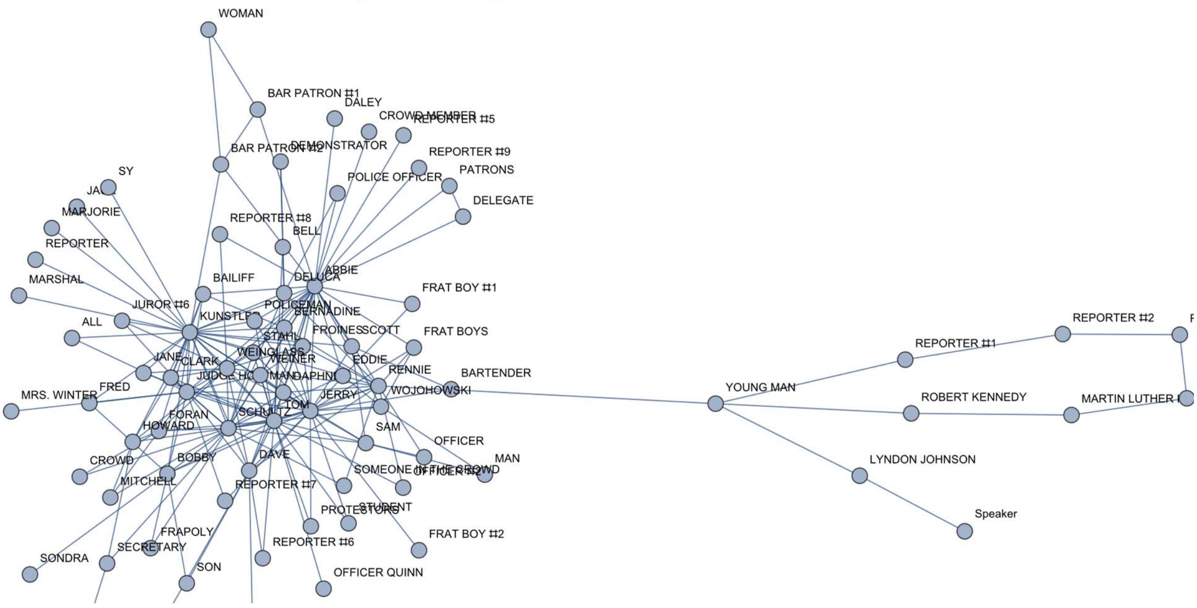
g. At first, we computed two weighted graphs – directed and undirected. We did so using the program Mathematica, as shown in the lecture. We then computed the unweighted graphs by removing all self-loops and double edged.

- **figure for the all 4 graph:**
  ***** Those figure can found in the Mathematica .nb file and as jpg file.

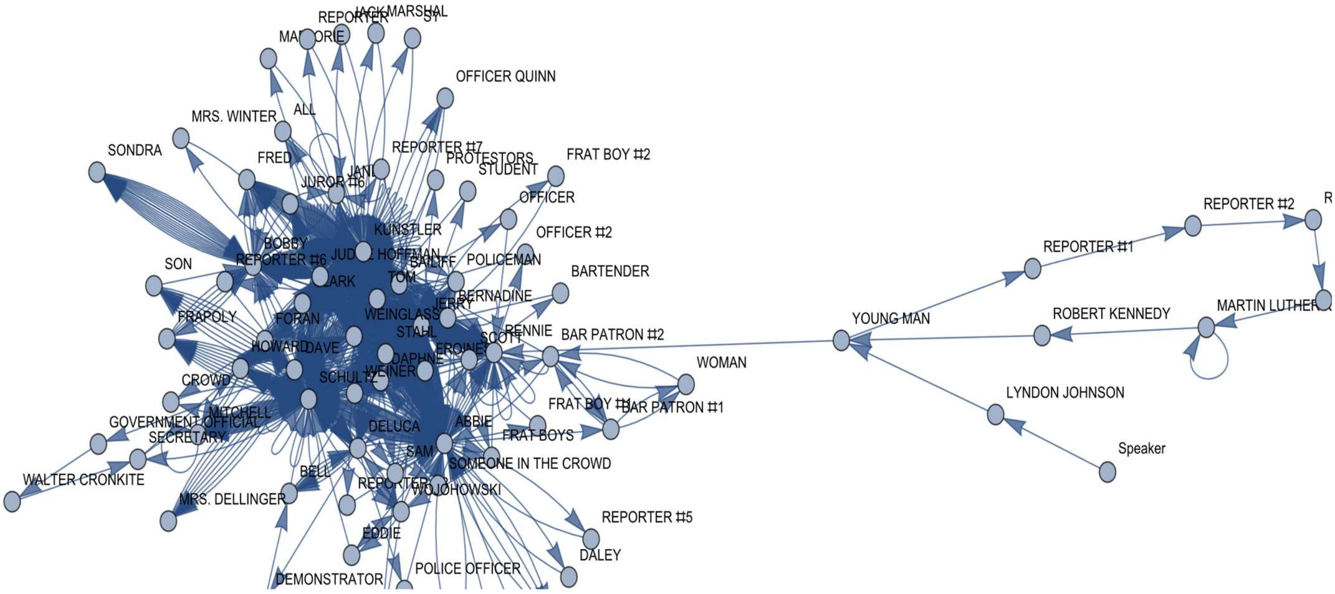
1. UndirectedWeightedGraph1



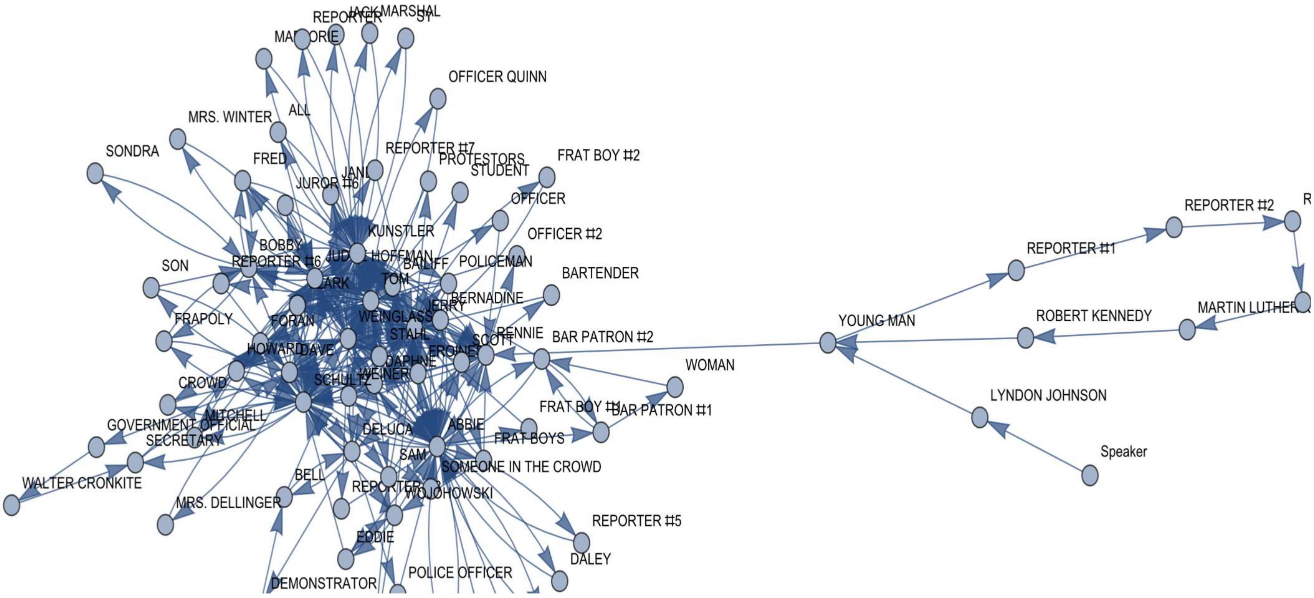
2. UndirectedNonWeightedGraph2



3. DirectedWeightedGraph3



4. DirectedNonWeightedGraph4



h. **Degree Centrality:**

We run kind of this command to each Graph: (to show it already sort)

```
SortBy[Transpose[{VertexList[UndirectedWeightedGraph1], DegreeCentrality[UndirectedWeightedGraph1]}], Last]
```

For graphs 1-2:
Abbie:30, Kunstler:30, Jerry:24, Tom:23

For graphs 3-4:
Kunstler: 54, Abbie:48, Jerry: 39, Schultz:38

i. **Page Rank Centrality**

We run kind of this command to each Graph: (to show it already sort)

```
SortBy[Transpose[{VertexList[UndirectedWeightedGraph1], PageRankCentrality[UndirectedWeightedGraph1, 0.85]}], Last];
```

When running page rank centrality, we got these results:

For UndirectedWeightedGraph1

Kunstler: 0.11, Tom: 0.09, Abbie: 0.078, Judge Hoffman: 0.069

For UndirectedNonWeightedGraph2

Abbie: 0.0687, Kunstler: 0.0641, Jerry: 0.0507, Tom: 0.04978

For DirectedWeightedGraph3:

Kunstler: 0.11, Tom:0.0948, Abbie: 0.0790, Judge Hoffman: 0.0740

For DirectedNonWeightedGraph4:

Kunstler: 0.11, Tom:0.0948, Jerry:0.059, Tom:0.0564

j. **Closeness Centrality**

We run kind of this command to each Graph: (to show it already sort)

```
SortBy[Transpose[{VertexList[UndirectedWeightedGraph1], ClosenessCentrality[UndirectedWeightedGraph1]}], Last];
```

For UndirectedWeightedGraph1:

Kunstler: 0.553, Abbie: 0.5531, Tom: 0.5306, Jerry: 0.5306

For UndirectedNonWeightedGraph2:

Kunstler: 0.553, Abbie: 0.551, Tom: 0.530, Jerry: 0.5306

For DirectedWeightedGraph3:

Abbie: 0.58, Kunstler: 0.57, Schultz: 0.542, Tom: 0.534

For DirectedNonWeightedGraph4:

Abbie: 0.58, Kunstler: 0.573, Schultz: 0.542, Tom: 0.534

Part 2

a. We searched online and found the srt file

b. We wrote a python script that converts the script into csv. Similar to 1-f, we realized the patter in the file, parsed the begin and end time, grabbed the text that was said and pushed all into a csv file. Then we wrote a program to merge between the csv files. The programs logic was as followed: going line by line in the script, we search for the line that matches best a line in the srt (using built in library difflib). If found, we grab that speaker, text, and times, and append to list. If not, we split the script into four and search each one separately. At this point, we are highly likely to match to a srt. We then sorted by time of speech, and found ourselves with a beautiful merge-srt.csv file.
The code file is attached called *"merge.py*" (and it used *"line.py*" too)

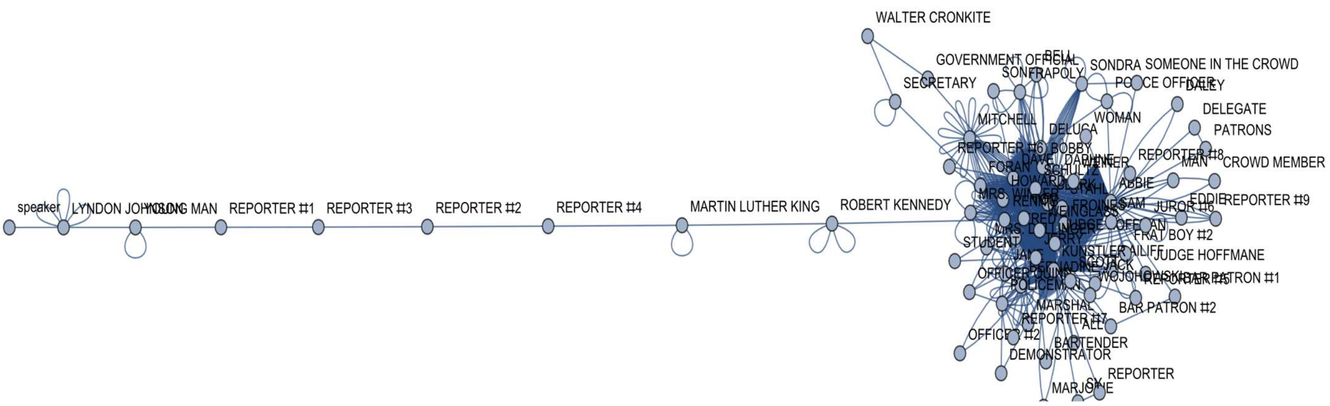
c. Compute the AB-SRT-graph. (Above)

d-e. (Same, according to Pro. Zvi)

<u>figure for the all 4 graph:</u>

* Those figure can found in the Mathematica .nb file and as jpg file.

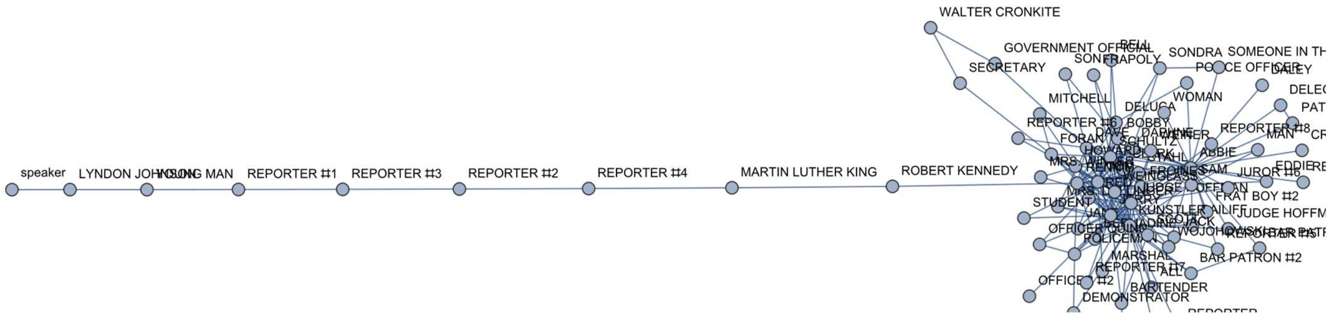
1. **For UndirectedWeightedGraph1**



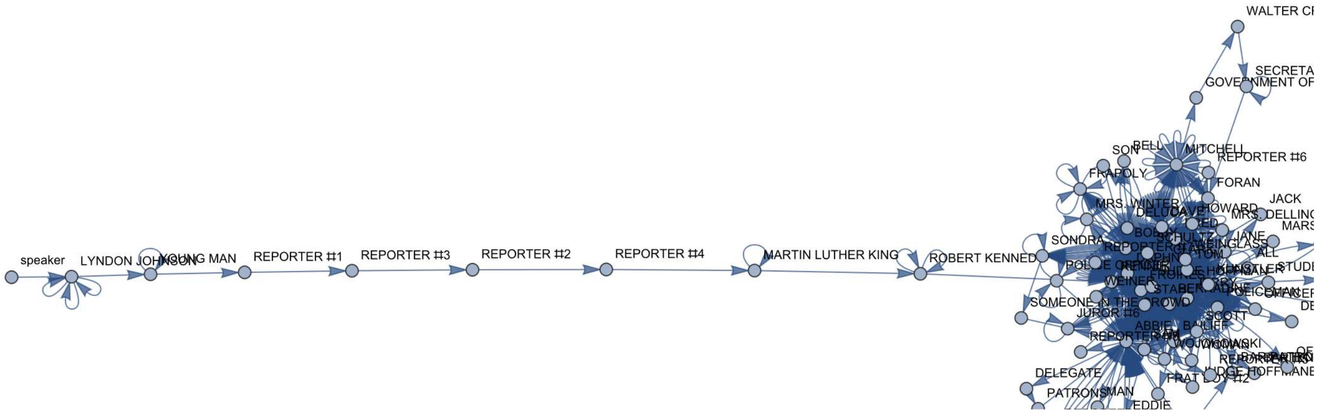
(Maybe it refer as to the "Evil" and bad things we had at the begingg on the movie that it talked there about The Vietnam war, and about the Viloece Proteset that was in U.S on those Times. As we can see in Episode 21 in the book " Evil in Social Networks" (there page 338)[1]

[1] Analyzing Narratives in Social Networks Taking Turing to the Arts by Zvi Lotker(Chapter 21, page 333-344)

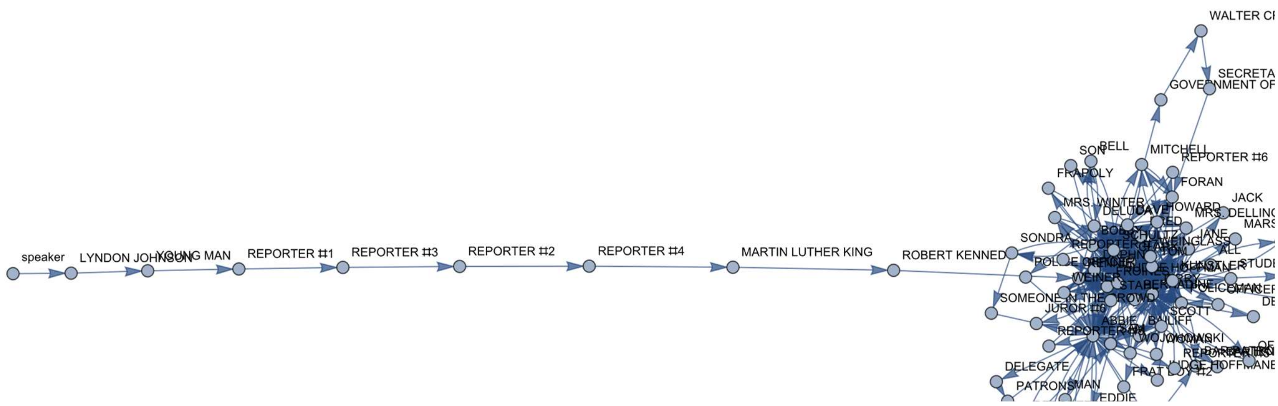
2. **UndirectedNonWeightedGraph2**



3. DirectedWeightedGraph3



4. DirectedNonWeightedGraph4



Part 3

See 2b-

Part 4

a. **The main conflict**:
The main conflict in the movie is between the prosecution – the U.S. represented by Schultz, Vs. the defendants and their lawyer. Although the job of a judge in a court of justice to be unbiased, it is clear that Judge Hoffman is on team Schultz.
In addition, we can add that within the defendants there is another conflict between Abbie and Tom. This conflict is regarding how to address the situation.

b.
Team A: Richard Schultz, Judge Hoffman
Team B: The Chicago 7 and their lawyer

c. **Anchors\Leader**

Team A: Schultz
Team B: Tom Hayden and Abbie Hoffman
Since Schultz is representing the state, the defendants needed two anchors so that they'd have the same strength. In addition, it is clear that within this party there are two different anchors.

d. **online references that support our conflicts.**
https://www.townandcountrymag.com/leisure/arts-and-culture/a34361792/the-trial-of-the-chicago-7-1968-democratic-convention-true-story/

https://en.wikipedia.org/wiki/The_Trial_of_the_Chicago_7

Part 5

Create the AB graph with the eight main characters in your movie/script:

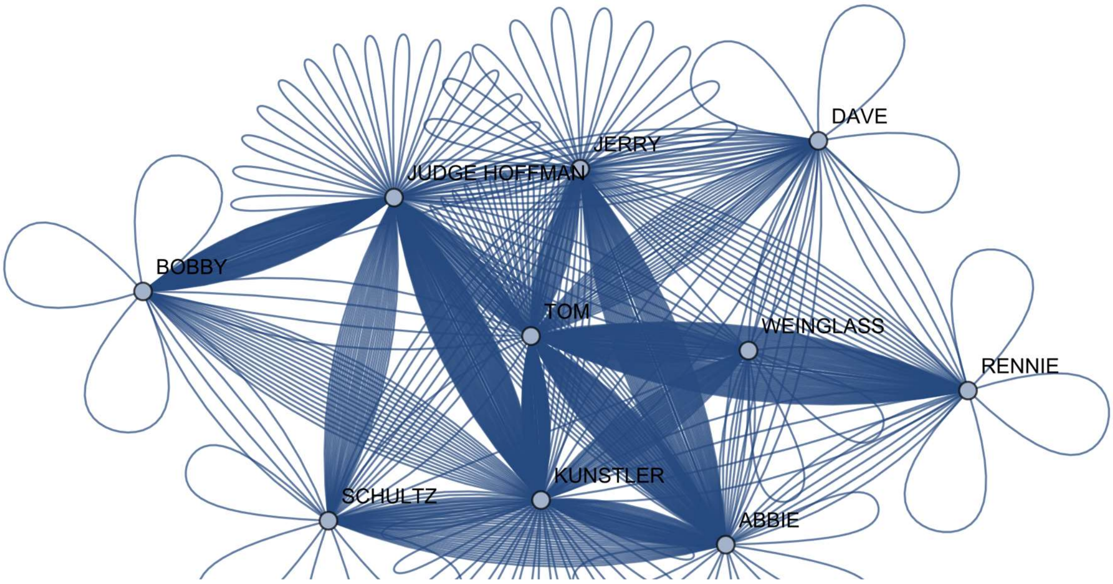
We use those command:
(we do It already for the Undirect Graph and for the Direct Graph)

```
namesUnDirect = Take[SortBy[Transpose[{VertexList[UndirectedWeightedGraph1], VertexDegree[UndirectedWeightedGraph1]}], Last], {-10, -1}][[;; , 1]];
namesDirect = Take[SortBy[Transpose[{VertexList[DirectedWeightedGraph3], VertexDegree[DirectedWeightedGraph3]}], Last], {-10, -1}][[;; , 1]]
{DAVE, WEINGLASS, RENNIE, BOBBY, JERRY, SCHULTZ, ABBIE, JUDGE HOFFMAN, TOM, KUNSTLER}
```

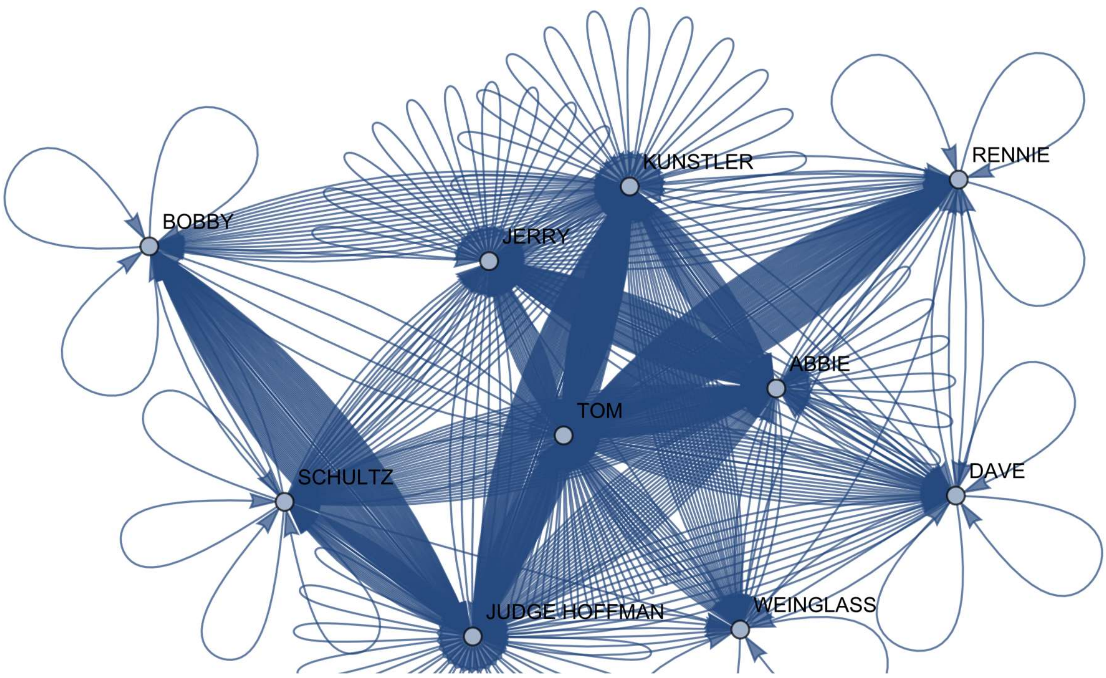
```
In[78]:= TopgGrapUndirect1 = Subgraph[UndirectedWeightedGraph1, namesUnDirect, VertexLabels → "Name"]
TopgGrapDirect3 = Subgraph[DirectedWeightedGraph3, namesDirect, VertexLabels → "Name"]
```

Now we can see more visible the graph with the main characters:

<u>TopgGrapUndirect1</u>:



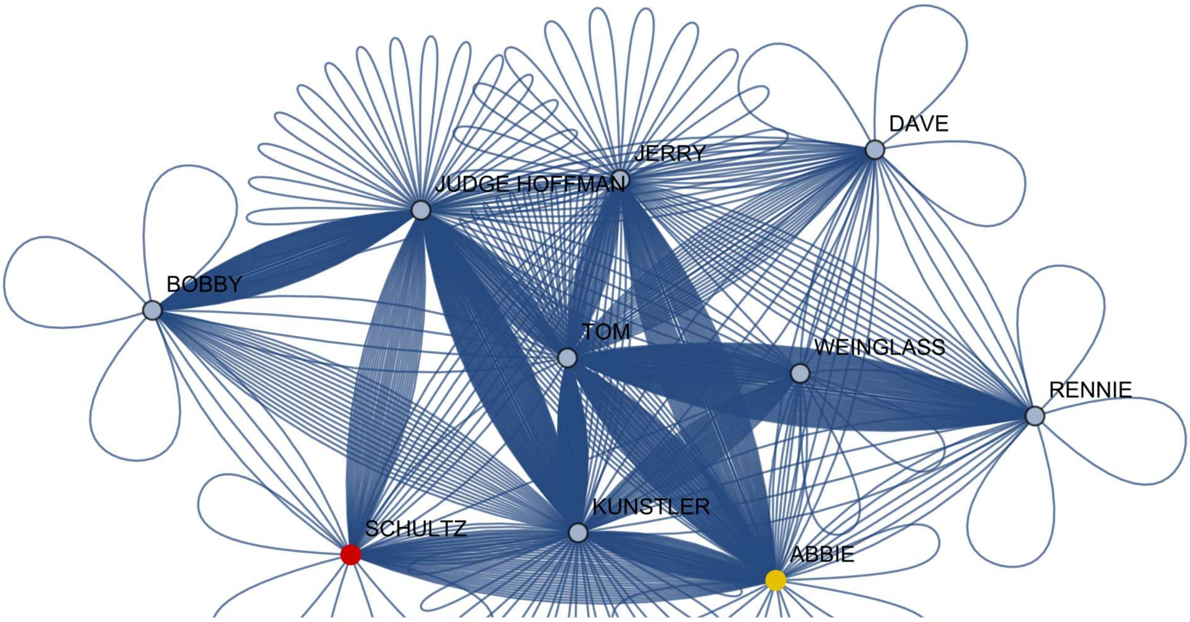
<u>TopgGrapDirect3:</u>



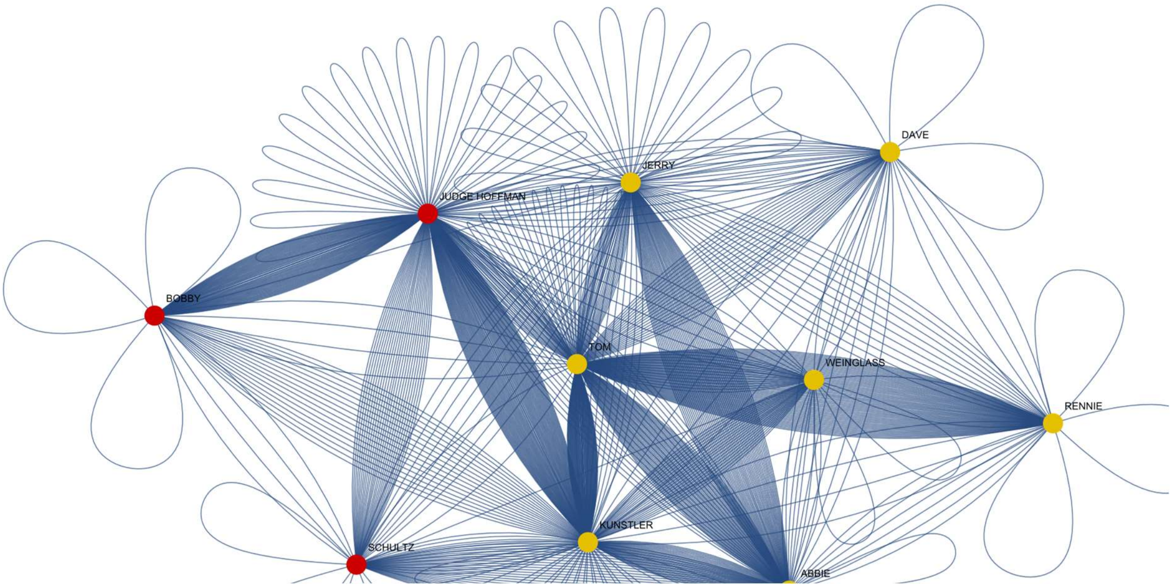
a. **Voronoi algorithm**:

At first, we moved our graph to be a distance matrix be setting distance between non-adjacent vertices to ∞, to adjacent vertices by 1/degree, and obviously distance to itself as 0. Then we ran a function to turn this distance to a metric space.
Once we have our matrix, we asked every character to which party is he closer to – and with that we managed to split all the characters to different parties.

The Graph with the anchors colored:



The Graph with the parties colored by this algorithm:



b. <u>**Voting algorithm**</u>

For this algorithm, we also used a distance matrix. In order to create to decision matrix, we set the vectors representing the anchors to choose only themselves. Once we had this, we multiplied the decision matrix by itself 100 times to ensure it's convergence. Now, all that was left was to take every character from the 10 that we chose and ask him to choose an anchor – by multiplying him (from the left) by the matrix. Of the two values in the output, we choose the smaller one, as that one is the closer anchor.

VOTING ALGO

We can transform a Frequency graph to a Markov process by simple normalize the Frequency (on TopgGrapDirect3)

In[110]:=
```
FromGtoMarkve[G_] := Module[{WA, M}, (*The input is a Grapg G The output is Transition Matrix M*)WA = WeightedAdjacencyMatrix[G];
   (*Converts the graph in to weighted Adjacency matrix*)M = # / Total[#] & /@ Normal[WA] (*Converts the weighted Adjacency matrix in to transition Markov matrix*)];
```

In[111]:=
```
FromGtoMarkve[TopgGrapDirect3]
MatrixForm[%];
```

Out[111]=

$$\{\{\tfrac{3}{29}, \tfrac{2}{29}, \tfrac{2}{29}, 0, \tfrac{5}{29}, 0, \tfrac{2}{29}, \tfrac{4}{29}, \tfrac{9}{29}, \tfrac{2}{29}\}, \{\tfrac{1}{25}, \tfrac{2}{25}, 0, 0, \tfrac{1}{25}, 0, \tfrac{2}{25}, \tfrac{3}{25}, \tfrac{7}{25}, \tfrac{9}{25}\}, \{\tfrac{2}{47}, \tfrac{2}{47}, \tfrac{3}{47}, 0, \tfrac{7}{47}, 0, \tfrac{3}{47}, 0, \tfrac{28}{47}, \tfrac{2}{47}\},$$
$$\{0, 0, 0, \tfrac{3}{47}, \tfrac{1}{47}, 0, 0, \tfrac{32}{47}, \tfrac{2}{47}, \tfrac{9}{47}\}, \{\tfrac{3}{62}, 0, \tfrac{2}{31}, 0, \tfrac{11}{62}, \tfrac{3}{62}, \tfrac{11}{31}, \tfrac{3}{62}, \tfrac{5}{31}, \tfrac{3}{31}\}, \{0, \tfrac{1}{40}, 0, \tfrac{1}{10}, \tfrac{1}{8}, \tfrac{1}{10}, \tfrac{11}{40}, \tfrac{9}{40}, \tfrac{1}{40}, \tfrac{1}{8}\}, \{\tfrac{4}{91}, \tfrac{3}{91}, \tfrac{3}{91}, 0, \tfrac{19}{91}, \tfrac{16}{91}, \tfrac{6}{91}, \tfrac{1}{13}, \tfrac{18}{91}, \tfrac{15}{91}\},$$
$$\{\tfrac{6}{119}, \tfrac{3}{119}, 0, \tfrac{27}{119}, \tfrac{3}{119}, \tfrac{16}{119}, \tfrac{12}{119}, \tfrac{16}{119}, \tfrac{9}{119}, \tfrac{27}{119}\}, \{\tfrac{9}{139}, \tfrac{7}{139}, \tfrac{31}{139}, \tfrac{2}{139}, \tfrac{10}{139}, \tfrac{1}{139}, \tfrac{19}{139}, \tfrac{10}{139}, \tfrac{13}{139}, \tfrac{37}{139}\}, \{\tfrac{1}{65}, \tfrac{1}{13}, \tfrac{1}{65}, \tfrac{4}{65}, \tfrac{3}{130}, \tfrac{3}{65}, \tfrac{6}{65}, \tfrac{37}{130}, \tfrac{19}{65}, \tfrac{6}{65}\}\}$$

Here we change manually vector 6-7 which are our anchors

In[113]:= MatDecsion =

$$\{\{\tfrac{3}{29}, \tfrac{2}{29}, \tfrac{2}{29}, 0, \tfrac{5}{29}, 0, \tfrac{2}{29}, \tfrac{4}{29}, \tfrac{9}{29}, \tfrac{2}{29}\}, \{\tfrac{1}{25}, \tfrac{2}{25}, 0, 0, \tfrac{1}{25}, 0, \tfrac{2}{25}, \tfrac{3}{25}, \tfrac{7}{25}, \tfrac{9}{25}\}, \{\tfrac{2}{47}, \tfrac{2}{47}, \tfrac{3}{47}, 0, \tfrac{7}{47}, 0, \tfrac{3}{47}, 0, \tfrac{28}{47}, \tfrac{2}{47}\},$$
$$\{0, 0, 0, \tfrac{3}{47}, \tfrac{1}{47}, 0, 0, \tfrac{32}{47}, \tfrac{2}{47}, \tfrac{9}{47}\}, \{\tfrac{3}{62}, 0, \tfrac{2}{31}, 0, \tfrac{11}{62}, \tfrac{3}{62}, \tfrac{11}{31}, \tfrac{3}{62}, \tfrac{5}{31}, \tfrac{3}{31}\}, \{0, 0, 0, 0, 0, 1, 0, 0, 0, 0\}, \{0, 0, 0, 0, 0, 0, 1, 0, 0, 0\},$$
$$\{\tfrac{6}{119}, \tfrac{3}{119}, 0, \tfrac{27}{119}, \tfrac{3}{119}, \tfrac{16}{119}, \tfrac{12}{119}, \tfrac{16}{119}, \tfrac{9}{119}, \tfrac{27}{119}\}, \{\tfrac{9}{139}, \tfrac{7}{139}, \tfrac{31}{139}, \tfrac{2}{139}, \tfrac{10}{139}, \tfrac{1}{139}, \tfrac{19}{139}, \tfrac{10}{139}, \tfrac{13}{139}, \tfrac{37}{139}\}, \{\tfrac{1}{65}, \tfrac{1}{13}, \tfrac{1}{65}, \tfrac{4}{65}, \tfrac{3}{130}, \tfrac{3}{65}, \tfrac{6}{65}, \tfrac{37}{130}, \tfrac{19}{65}, \tfrac{6}{65}\}\}$$

Out[113]=

$$\{\{\tfrac{3}{29}, \tfrac{2}{29}, \tfrac{2}{29}, 0, \tfrac{5}{29}, 0, \tfrac{2}{29}, \tfrac{4}{29}, \tfrac{9}{29}, \tfrac{2}{29}\}, \{\tfrac{1}{25}, \tfrac{2}{25}, 0, 0, \tfrac{1}{25}, 0, \tfrac{2}{25}, \tfrac{3}{25}, \tfrac{7}{25}, \tfrac{9}{25}\}, \{\tfrac{2}{47}, \tfrac{2}{47}, \tfrac{3}{47}, 0, \tfrac{7}{47}, 0, \tfrac{3}{47}, 0, \tfrac{28}{47}, \tfrac{2}{47}\},$$
$$\{0, 0, 0, \tfrac{3}{47}, \tfrac{1}{47}, 0, 0, \tfrac{32}{47}, \tfrac{2}{47}, \tfrac{9}{47}\}, \{\tfrac{3}{62}, 0, \tfrac{2}{31}, 0, \tfrac{11}{62}, \tfrac{3}{62}, \tfrac{11}{31}, \tfrac{3}{62}, \tfrac{5}{31}, \tfrac{3}{31}\}, \{0, 0, 0, 0, 0, 1, 0, 0, 0, 0\}, \{0, 0, 0, 0, 0, 0, 1, 0, 0, 0\},$$
$$\{\tfrac{6}{119}, \tfrac{3}{119}, 0, \tfrac{27}{119}, \tfrac{3}{119}, \tfrac{16}{119}, \tfrac{12}{119}, \tfrac{16}{119}, \tfrac{9}{119}, \tfrac{27}{119}\}, \{\tfrac{9}{139}, \tfrac{7}{139}, \tfrac{31}{139}, \tfrac{2}{139}, \tfrac{10}{139}, \tfrac{1}{139}, \tfrac{19}{139}, \tfrac{10}{139}, \tfrac{13}{139}, \tfrac{37}{139}\}, \{\tfrac{1}{65}, \tfrac{1}{13}, \tfrac{1}{65}, \tfrac{4}{65}, \tfrac{3}{130}, \tfrac{3}{65}, \tfrac{6}{65}, \tfrac{37}{130}, \tfrac{19}{65}, \tfrac{6}{65}\}\}$$

Multiply matrix by a vector that represent character
we do it for each character ans see what he choose to "vote" to

Most of the answer you can find on the notebook, we do it for each character:
Like those:

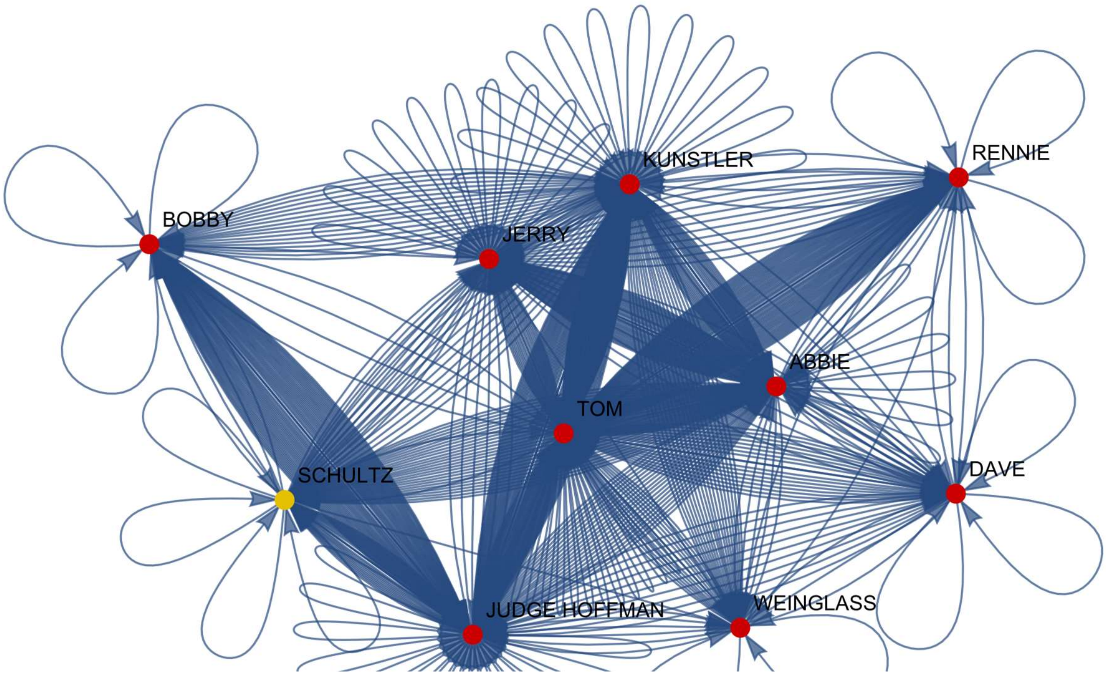
In[118]:=
```
(* 1. Dave choose  -- > ABBIE *)
{1, 0, 0, 0, 0, 0, 0, 0, 0, 0}.MatrixPower[MatDecsion, 100] // N
```

Out[118]= $\{2.18479\times10^{-9}, 2.23119\times10^{-9}, 3.43546\times10^{-9}, 3.49754\times10^{-9}, 3.2843\times10^{-9}, 0.230343, 0.769657, 8.97014\times10^{-9}, 9.76529\times10^{-9}, 8.9502\times10^{-9}\}$

In[119]:=
```
(*2. WEINGLASS choose  -- > ABBIE*)
{0, 1, 0, 0, 0, 0, 0, 0, 0, 0}.MatrixPower[MatDecsion, 100] // N
```

Out[119]= $\{2.27226\times10^{-9}, 2.32051\times10^{-9}, 3.57299\times10^{-9}, 3.63756\times10^{-9}, 3.41578\times10^{-9}, 0.251706, 0.748294, 9.32924\times10^{-9}, 1.01562\times10^{-8}, 9.30851\times10^{-9}\}$

Another way from lecture to represent VOTING result with the implemented Votkp func:



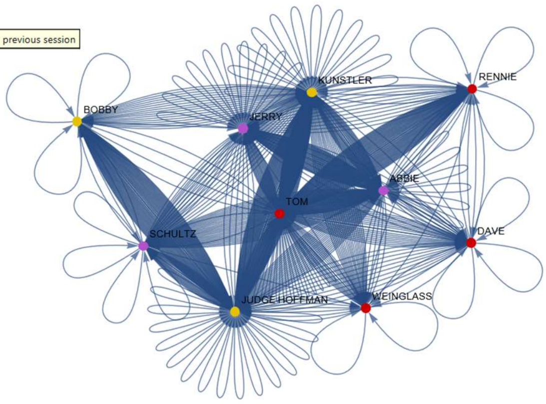
c. For the most part the algorithms gave the same result, other than **Bobby**. With Bobby, we found that in the Voronoi algorithm he chose Schultz, and with Voting algorithm he chose the guys from the Chicago 7. This adds up and shows how well these algorithms describe the movies. When asking to Bobby to choose a character that he's most in contact with – so the answer is the Judge, as his case really doesn't have anything to do with the other Chicago 7. This is computed accurately by Voronoi. However, when asked to choose a team by who he feels closest with – he will obviously choose the Chicago 7, and not the representatives of the government. This is computed accurately by Voring algorithm.

d. The figure for all of those algorithms are appear in those order.

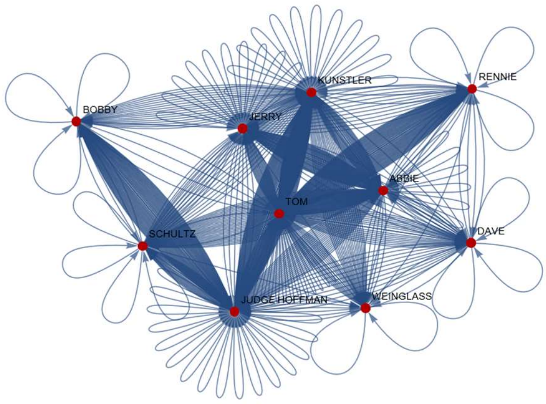
```
In[]:= options = {"Modularity", "Centrality", "CliquePercolation", "Hierarchical", "Spectral", "VertexMoving"};
```

Out[132] in a previous session

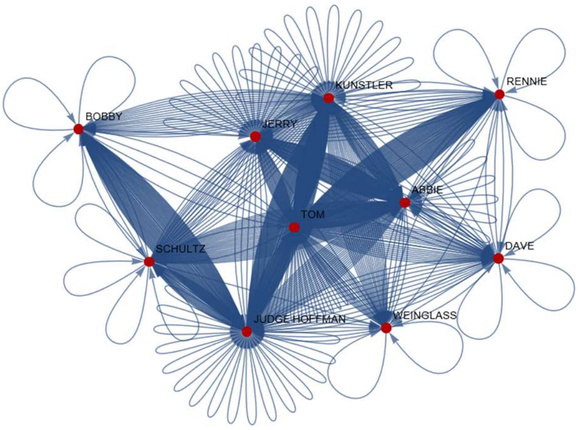
Out[]= {



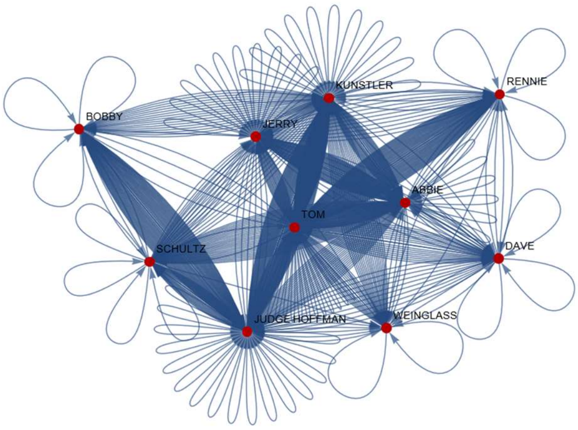
,



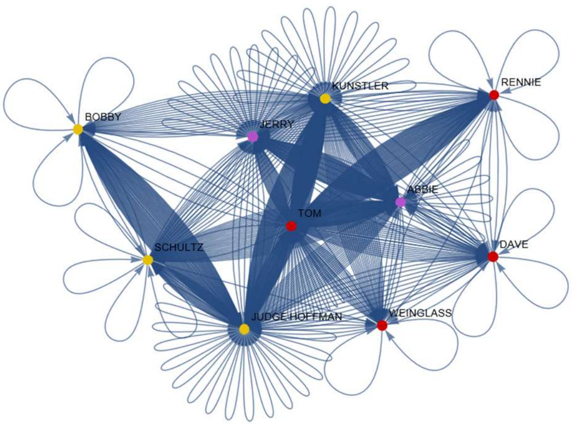
,



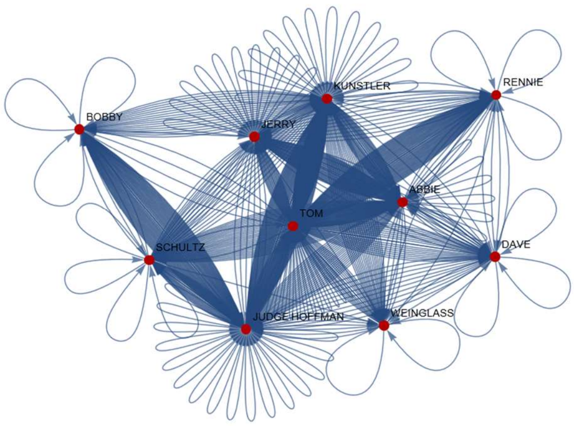
,



,



,



}

Part 6

a. As mentioned above, we found that the algorithms do not agree regarding Bobby.
b. This disagreement can be explained when understanding the movie in depth. On the one hand Bobby is in line with the Chicago 7 – he too is protesting the government, and he too is on trial for acts that he has done. However, his story is not the same as their story. While they are on trial for rioting, he is on trial for a speech he gave. In addition, it is obvious that he is getting far worse mistreated, probably because he is black and is the leader of the Black Panthers.
c. Sun Tzu:
    i. The Voronoi represents on indirect attack. The characters are divided to two groups, but they are each time asked about the anchors in general. On contrary, in voting we first compute the transition matrix, and create black holes for the anchors, and compute the algorithm using this matrix. This is a direct attack.
    ii. Computed in notebook
    iii. Computed in notebook
    iv. When discussing the matrix $\psi_{com} \quad (t): L_V^a \times \mathcal{M}_{n,n}[\mathbb{R}] \rightarrow \mathcal{M}_{n,k}[\mathbb{R}]$, which is linear combination of the matrices we have been discussing, we realise that for any $t \in [0,1]$, the output of the linear transformation will describe relationships between the characters, based on the inputs. That is, we ask every character to choose a party/anchor according to both algorithms, and then average between them with the weight t. As such, we will get a grouping to anchors. We can determine on which algorithm to put more of an emphasize by dialing t up from 0 to 1
d. Since we chose two parties with very difference sizes (government 2, convicts 8) – we didn't see relevance to look for a t to have parties of the same size, since by essence they're not the same size. Instead, we went along with our fascinating finding regarding Bobby. We found the t that caused Bobby to be same distance from both parties. Calculation in notebook

Part 7

In this section, we were asked to find the best conflict between the four main characters.

a. Here, we found the best conflict using entropy. We ran through all 14 possibilities of choosing either two-on-two or three-on-one and ran the voting algorithm. The results and explanations are in the notebook.
b. In this question, we found the best conflicts using variance. To do so, for each group we found how far every character is from his group, and by that we could determine which conflict is best. Here two, the algorithm results and explanations can be found in notebook.
c. In both cases, we found that the results matched what our human perception of the movie understood to be the vest conflict. So cool!!