Elaboration d'un graphe en profondeur

Choix de hubs subjectifs de départ : Le Figaro, qui est de "droite gaulliste, libérale et conservatrice". Libération, qui est "un journal libertaire de gauche"

1. Initialisation de Tweepy

In [1]:

```
import tweepy
```

In [2]:

```
from tweepy import OAuthHandler
```

In [3]:

```
consumer_key = 'L2mPv8Oxjs2mLtBEWZZiyDy0W'
consumer_secret = 'idPHQogjNLSKdo7ns8exk9ZXSbIsqcd4UsXd1vofliwj1s2iZj'
access_token = '1413867470-bWJRTSz2Ece90tG1Z5IU9pmPZQvdWR10xrWqNBL'
access_secret = '6Z0X9eJ0dLlYJj8JrqX6VcNMOOsTWpaxaFosJhiv5Xkoa'

auth = OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_secret)

api = tweepy.API(auth, wait_on_rate_limit=True, wait_on_rate_limit_notify=True, compression=True)
```

1. Initialisation des hubs

In [4]:

```
hub1 = api.get_user('Le_Figaro')
hub2 = api.get_user('libe')
print(hub1.screen_name)
print(hub1.followers_count)
print('-----------')
print(hub2.screen_name)
print(hub2.followers_count)
```

```
Le_Figaro
3340630
-----------
libe
3182480
```

1. Récupération de quelques comptes au hasard parmi les 600 000 (pas besoin de tous les prendre, la limite de twitter va nous limiter à ne parcourir les abonnements que de quelques centaines d'utilisateurs)

In [5]:

```
def limit_handled(cursor):
    while True:
        try:
            yield cursor.next()
        except tweepy.RateLimitError:
            time.sleep(15 * 60)
```

In [8]:

```
import pandas as pd
import numpy as np
import csv
```

In [73]:

```
followersids_hub1 = []
followersids_hub2 = []
```

In [74]:

```
with open("followersid_hub1.csv", newline="") as f:
  csvreader = csv.reader(f)
  for row in csvreader:
    i = int(row[0])
    followersids_hub1.append(i)
with open("followersid_hub2.csv", newline="") as f:
  csvreader = csv.reader(f)
  for row in csvreader:
    i = int(row[0])
    followersids_hub2.append(i)
```

In [95]:

```
users_hub1 = []
for i in range(50):
    fin_cycle = min((i + 1) * 50, len(followersids_hub1))
    users_hub1.extend(
        api.lookup_users(user_ids=followersids_hub1[i * 50:fin_cycle])
    )
```

In [96]:

```
users_hub2 = []
for i in range(50):
    fin_cycle = min((i + 1) * 50, len(followersids_hub2))
    try:
        users_hub2.extend(
            api.lookup_users(user_ids=followersids_hub2[i * 50:fin_cycle])
        )
    except:
        print('User suivant')
```

In [97]:

```
# On retire les comptes doublons
print(len(users_hub1))
print(len(users_hub2))
for j in users_hub1:
    for i in users_hub2:
        if i.screen_name == j.screen_name:
            users_hub1.remove(i)
print(len(users_hub1))
print(len(users_hub2))
```

```
2493
2494
1821
2494
```

1. Récupération des abonnements de 340 followers des hubs (170 hub1, 170 hub2). Environ 4h30 de temps.

In [99]:

```
abonnementsids_hub1 = {}
for i in range(170):
    # Pour chaque utilisateur, on récupère ses abonnements avec la même méthode
    tmpids = []
    # Ne pas oublier la gestion d'erreur pour les comptes privés (jusque 13%), il y en a beaucoup
!!
    try:
        for abonnement in tweepy.Cursor(api.friends_ids, screen_name=users_hub1[i].screen_name, cou
```

```
nt=5000).items():
            tmpids.append(abonnement)
    except tweepy.TweepError:
        print("Erreur utilisateur, au suivant")
    abonnementsids_hub1[users_hub1[i].screen_name] = tmpids
len(abonnementsids_hub1)
```

```
Rate limit reached. Sleeping for: 479
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 893
```

```
Erreur utilisateur, au suivant
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 893
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 893
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 893
```

```
Erreur utilisateur, au suivant
Erreur utilisateur, au suivant
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 893
Rate limit reached. Sleeping for: 892
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
Erreur utilisateur, au suivant
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 893
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 893
```

```
Erreur utilisateur, au suivant
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
```

Out[99]:

```
170
```

In [100]:

```
abonnementsids_hub2 = {}
for i in range(170):
```

```
    # Pour chaque utilisateur, on récupère ses abonnements avec la même méthode
    tmpids = []
    # Ne pas oublier la gestion d'erreur pour les comptes privés (jusque 13%), il y en a beaucoup
!!
    try:
        for abonnement in tweepy.Cursor(api.friends_ids, screen_name=users_hub2[i].screen_name, cou
nt=5000).items():
            tmpids.append(abonnement)
    except tweepy.TweepError:
        print("Erreur utilisateur, au suivant")
    abonnementsids_hub2[users_hub2[i].screen_name] = tmpids
len(abonnementsids_hub2)
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 893
Rate limit reached. Sleeping for: 893
Rate limit reached. Sleeping for: 892
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 892
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
```

```
Rate limit reached. Sleeping for: 892
```

```
Erreur utilisateur, au suivant
```

Out[100]:

```
170
```

In [101]:

```
print(len(abonnementsids_hub1))
print(len(abonnementsids_hub2))
```

```
170
170
```

In [102]:

```
# Nombre d'abonnement que l'on a dans les 300
```

```
res = 0
for k, v in abonnementsids_hub1.items():
    for j in range(len(v)):
        res = res + 1
for k, v in abonnementsids_hub2.items():
    for j in range(len(v)):
        res = res + 1
print(res)
```

```
54986
```

In [103]:

```
df1 = pd.DataFrame.from_dict(abonnementsids_hub1, orient ='index')
df1.to_csv('abonnementsids_hub1.csv', sep =',')
df2 = pd.DataFrame.from_dict(abonnementsids_hub2, orient ='index')
df2.to_csv('abonnementsids_hub2.csv', sep =',')
```

1. Partie algorithmique : déterminer les liens entre ces abonnements pour définir les hubs

In [104]:

```
abonnementsids_final = {}
for key, value in abonnementsids_hub1.items():
        if len(value) > 0:
            abonnementsids_final[key] = value;
print(len(abonnementsids_hub1))
print(len(abonnementsids_final))

res = 0
for k, v in abonnementsids_final.items():
    for j in range(len(v)):
        res = res + 1
print(res)

abonnementsids_final_2 = {}
for key, value in abonnementsids_hub2.items():
        if len(value) > 0:
            abonnementsids_final_2[key] = value;
print(len(abonnementsids_hub2))
print(len(abonnementsids_final_2))

res = 0
for k, v in abonnementsids_final_2.items():
    for j in range(len(v)):
        res = res + 1
print(res)
```

```
170
153
23919
170
158
31067
```

In [105]:

```
def existe(cle, liste):
    if cle in liste:
        return "true"
    return "false"
```

In [106]:

```
liens = {}
liste_visite_ids = []
for k, v in abonnementsids_final.items():
    nbLiens = 0
    for i in range(len(v)):
        for key, value in abonnementsids_final.items():
            for i in range(len(value)):
```

```
            for j in range(len(value)):
                if(v[i] == value[j] and k != key and existe(key, liste_visite_ids) == "false"):
                    nbLiens = nbLiens + 1
        if(existe(v[i], liens) == "true"):
            liens[v[i]] += nbLiens
        else:
            liens[v[i]] = nbLiens
    liste_visite_ids.append(k)
print(len(liens))

liens_2 = {}
liste_visite_ids_2 = []
for k, v in abonnementsids_final_2.items():
    nbLiens = 0
    for i in range(len(v)):
        for key, value in abonnementsids_final_2.items():
            for j in range(len(value)):
                if(v[i] == value[j] and k != key and existe(key, liste_visite_ids_2) == "false"):
                    nbLiens = nbLiens + 1
        if(existe(v[i], liens_2) == "true"):
            liens_2[v[i]] += nbLiens
        else:
            liens_2[v[i]] = nbLiens
    liste_visite_ids_2.append(k)
print(len(liens_2))
```

```
17413
24026
```

Pour bien comprendre : on a les abonnements de plus de 300 utilisateurs, en tout 54 986 personnes. On dit que un lien, c'est un abonnement. Donc on compte le nombre de fois où un utilisateur apparait. On a alors 17 413 liens pour le hub 1, et 24 026 liens pour le hub 2.

In [134]:

```
# On tri les liens
liensTries = {}
liensTries_2 = {}
for k, v in sorted(liens.items(), key=lambda x: x[1], reverse=True):
    liensTries[k] = v
for k, v in sorted(liens_2.items(), key=lambda x: x[1], reverse=True):
    liensTries_2[k] = v
```

In [138]:

```
usersid_tries = []
usersid_tries_2 = []
lien_pondere = []
lien_pondere_2 = []
for k, v in liensTries.items():
    usersid_tries.append(k)
    lien_pondere.append(v)
for k, v in liensTries_2.items():
    usersid_tries_2.append(k)
    lien_pondere_2.append(v)
```

In [139]:

```
hubs = []
for i in range(int(len(usersid_tries)/100) + 1):
    fin_cycle = min((i + 1) * 100, len(usersid_tries))
    hubs.extend(
        api.lookup_users(user_ids=usersid_tries[i * 100:fin_cycle])
    )
```

In [140]:

```
hubs_2 = []
for i in range(int(len(usersid_tries_2)/100) + 1):
    fin_cycle = min((i + 1) * 100, len(usersid_tries_2))
    hubs_2.extend(
        api.lookup_users(user_ids=usersid_tries_2[i * 100:fin_cycle])
```

```
    )
```

In [141]:

```
# Concernant la communauté de Figaro
for i in range(len(hubs)):
    print("Nb liens : ", lien_pondere[i], " est ", hubs[i].screen_name, " avec ", hubs[i].followers
_count, " followers.")
```

```
Nb liens :  28983  est  EmmanuelMacron  avec  6405104  followers.
Nb liens :  28827  est  lemondefr  avec  9050772  followers.
Nb liens :  25285  est  Le_Figaro  avec  3340987  followers.
Nb liens :  21993  est  BarackObama  avec  126672313  followers.
Nb liens :  19043  est  JoeBiden  avec  19827537  followers.
Nb liens :  18175  est  le_Parisien  avec  2696324  followers.
Nb liens :  16872  est  RFI  avec  2678762  followers.
Nb liens :  16680  est  BFMTV  avec  2947381  followers.
Nb liens :  14854  est  France24_fr  avec  3594734  followers.
Nb liens :  14754  est  CNEWS  avec  1724464  followers.
Nb liens :  14476  est  franceinfo  avec  1753190  followers.
```

In [142]:

```
# Concernant la communauté de Libération
for i in range(len(hubs_2)):
    print("Nb liens : ", lien_pondere_2[i], " est ", hubs_2[i].screen_name, " avec ", hubs_2[i].fol
lowers_count, " followers.")
```

```
Nb liens :  39645  est  lemondefr  avec  9050778  followers.
Nb liens :  33825  est  EmmanuelMacron  avec  6405123  followers.
Nb liens :  31699  est  libe  avec  3182954  followers.
Nb liens :  28109  est  BarackObama  avec  126672429  followers.
Nb liens :  24371  est  le_Parisien  avec  2696324  followers.
Nb liens :  19995  est  franceinfo  avec  1753195  followers.
Nb liens :  19590  est  BFMTV  avec  2947384  followers.
Nb liens :  19481  est  JoeBiden  avec  19828095  followers.
Nb liens :  17914  est  CNEWS  avec  1724470  followers.
Nb liens :  17744  est  afpfr  avec  3571884  followers.
...
```

On créé les csv nécessaires pour le graphe

In [143]:

```
final_dico_figaro = {}
final_dico_libe = {}
for i in range(len(hubs)):
    final_dico_figaro[hubs[i].screen_name] = lien_pondere[i]
for i in range(len(hubs_2)):
    final_dico_libe[hubs_2[i].screen_name] = lien_pondere_2[i]
```

In [144]:

```
df3 = pd.DataFrame.from_dict(final_dico_figaro, orient ='index')
df3.to_csv('final_dico_figaro.csv', sep =',')
df4 = pd.DataFrame.from_dict(final_dico_libe, orient ='index')
df4.to_csv('final_dico_libe.csv', sep =',')
```

In [148]:

```
final_dico_unifie = {}
for d in final_dico_figaro, final_dico_libe:
    for key in d:
        final_dico_unifie[key] = final_dico_unifie.get(key, 0) + d[key]
len(final_dico_unifie)
```

Out[148]:

```
38342
```

In [150]:

```
df_final = pd.DataFrame.from_dict(final_dico_unifie, orient ='index')
df_final.to_csv('dico_final.csv', sep =',')
```

In [164]:

```
dico_figaro_user_abonnements = {}
for k, v in abonnementsids_final.items():
    new_t = []
    for i in range(int(len(v)/100) + 1):
        fin_cycle = min((i + 1) * 100, len(v))
        new_t.extend(
            api.lookup_users(user_ids=v[i * 100:fin_cycle])
        )
    dico_figaro_user_abonnements[k] = new_t
```

In [170]:

```
for k, v in dico_figaro_user_abonnements.items():
    nom = ''
    for i in range(len(v)):
        nom = v[i].screen_name
        v[i] = nom
```

In [173]:

```
dico_libe_user_abonnements = {}
for k, v in abonnementsids_final_2.items():
    new_t = []
    for i in range(int(len(v)/100) + 1):
        fin_cycle = min((i + 1) * 100, len(v))
        new_t.extend(
            api.lookup_users(user_ids=v[i * 100:fin_cycle])
        )
    dico_libe_user_abonnements[k] = new_t

for k, v in dico_libe_user_abonnements.items():
    nom = ''
    for i in range(len(v)):
        nom = v[i].screen_name
        v[i] = nom
```

In [171]:

```
df_final_figaro_abonnements = pd.DataFrame.from_dict(dico_figaro_user_abonnements, orient ='index'
)
df_final_figaro_abonnements.to_csv('dico_user_abos_fig.csv', sep =',')
```

In [174]:

```
df_final_libe_abonnements = pd.DataFrame.from_dict(dico_libe_user_abonnements, orient ='index')
df_final_libe_abonnements.to_csv('dico_user_abos_libe.csv', sep =',')
```

In [ ]:

```

```