Bitcoin network project

In [40]:
```
#Import liabraries
import pandas as pd
import numpy as np
import os
import csv
import pyarrow.parquet as pq
from operator import itemgetter
import networkx as nx
from IPython.display import Image
from IPython.core.display import HTML
```

In [18]:
```
#Set the directory
os.chdir('/Users/alina/Desktop/Bitcoin network/Data')
```

In [19]:
```
#Import data
#The two baseline periods representing bearish and bullish market w
#We have also included a month before the baseline period to see ho
df_jan19 = pd.read_parquet("part-2019-1-f6fd1362-3dce-44ad-8999-939
df_feb19 = pd.read_parquet("part-2019-2-77cbc838-f67c-4ad9-864e-40e
df_nov20 = pd.read_parquet("part-2020-11-e7988a0e-482b-43ed-8d0a-50
df_dec20 = pd.read_parquet("part-2020-12-83aa3ab8-7475-4768-941b-69
```

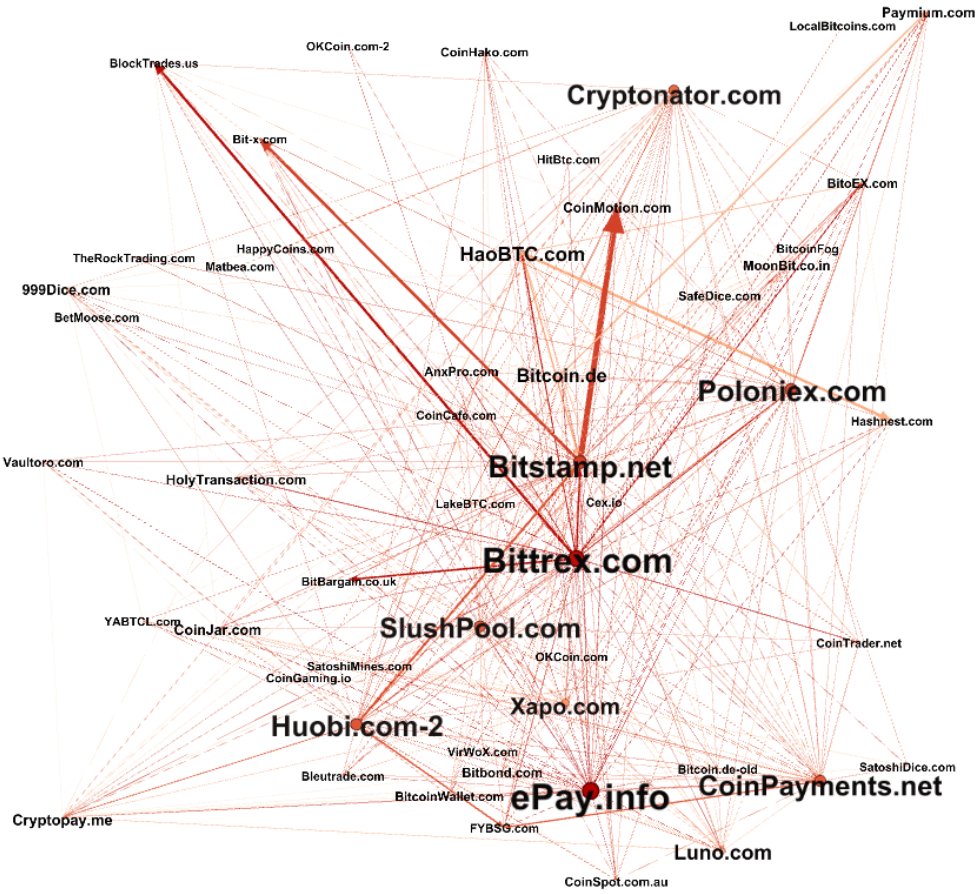
In [20]:
```
#Filter the data to have only identified users:
def filter_out_unique(df_dec20,src_identity=True,dst_identity=True)
    if src_identity:
        df_dec20 = df_dec20[df_dec20["src_identity"].str.len()<20]
    if dst_identity:
        df_dec20 = df_dec20[df_dec20["dst_identity"].str.len()<20]
    return df_dec20

def filter_out_anonym_cluster(df_dec20,src=True,dst=True):
    if src:
        df_dec20 = df_dec20[~df_dec20["src_identity"].astype(str).s
    if dst:
        df_dec20 = df_dec20[~df_dec20["dst_identity"].astype(str).s
    return df_dec20

def filter_out_self_spending(df_dec20):
    return df_dec20 [df_dec20 ["src_identity"]!=df_dec20 ["dst_iden

#Filtering for december 2020 data
df_dec20 = filter_out_unique(df_dec20)
df_dec20 =  filter_out_anonym_cluster(df_dec20)
df_dec20 = filter_out_self_spending(df_dec20)
#Filtering for november 2020 data
df_nov20 = filter_out_unique(df_nov20)
df_nov20 =  filter_out_anonym_cluster(df_nov20)
df_nov20 = filter_out_self_spending(df_nov20)
#Filtering for february 2019 data
df_feb19 = filter_out_unique(df_feb19)
df_feb19 =  filter_out_anonym_cluster(df_feb19)
df_feb19 = filter_out_self_spending(df_feb19)
#Filtering for january 2019 data
df_jan19 = filter_out_unique(df_jan19)
df_jan19 =  filter_out_anonym_cluster(df_jan19)
df_jan19 = filter_out_self_spending(df_jan19)
```

In [21]:
```
#Create the network on python
G_dec20 = nx.from_pandas_edgelist(df_dec20, 'src_identity', 'dst_id
G_nov20 = nx.from_pandas_edgelist(df_nov20, 'src_identity', 'dst_id
G_feb19 = nx.from_pandas_edgelist(df_feb19, 'src_identity', 'dst_id
G_jan19 = nx.from_pandas_edgelist(df_jan19, 'src_identity', 'dst_id
```

Graphs

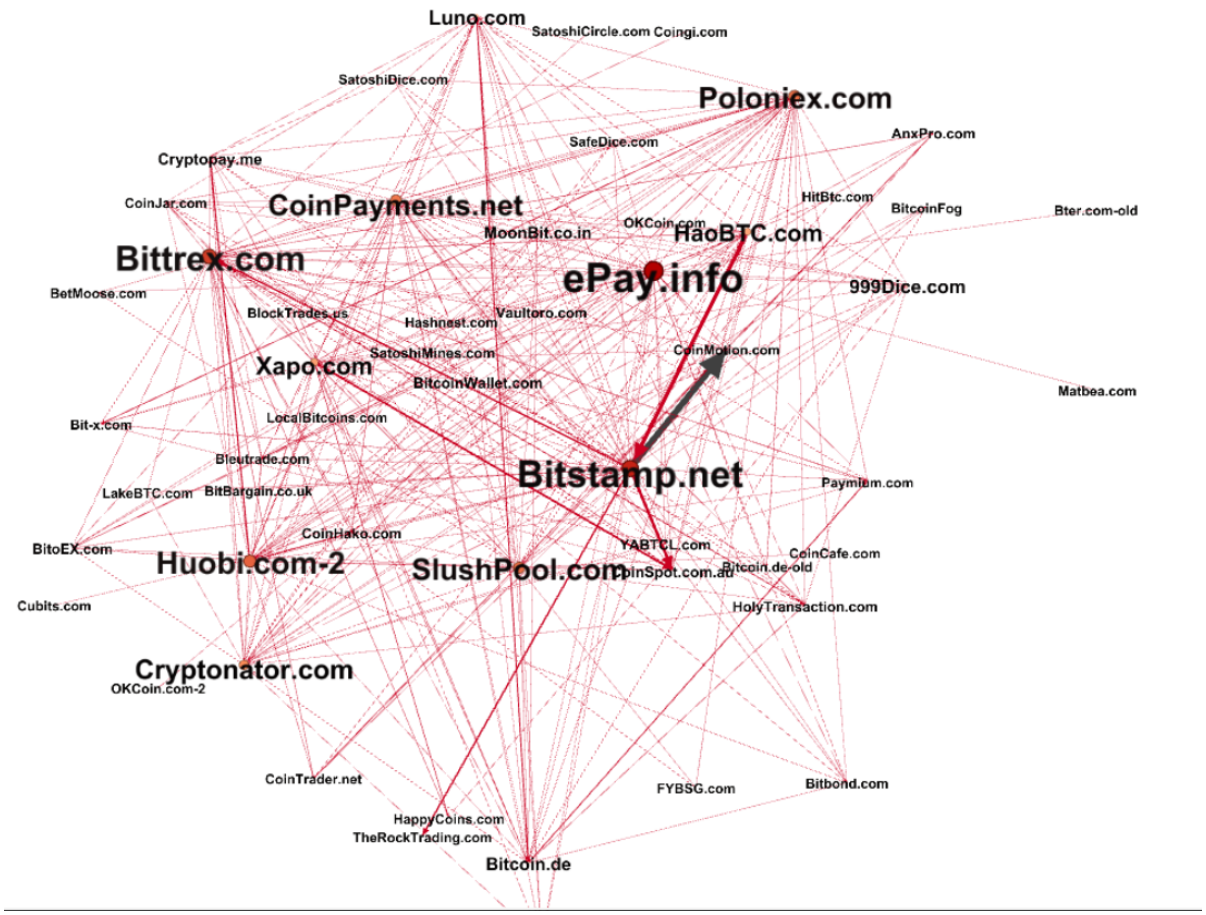
In [25]:
```
#To export a graph from python to Gephi with the networkx package (
#nx.write_gexf(G_dec20, 'dec20.gexf')
#nx.write_gexf(G_nov20, 'nov20.gexf')
#nx.write_gexf(G_feb19, 'feb20.gexf')
#nx.write_gexf(G_jan19, 'jan20.gexf')

#To display the graphs jan19 in Jupyter
Image(filename = "jan19.PNG", width=700, height=100)
```

Out[25]:



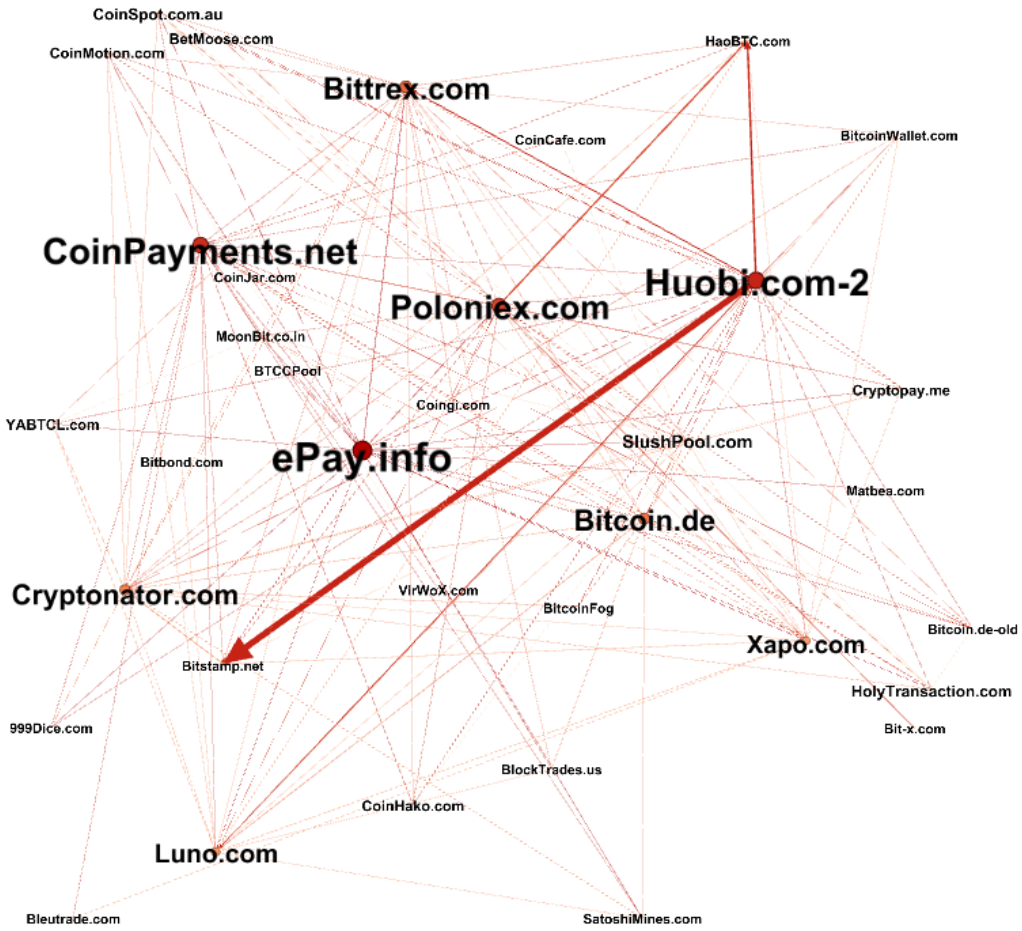
In [23]:
```
#To display the graphs feb19 in Jupyter
Image(filename = "feb19.PNG", width=700, height=100)
```

Out[23]:



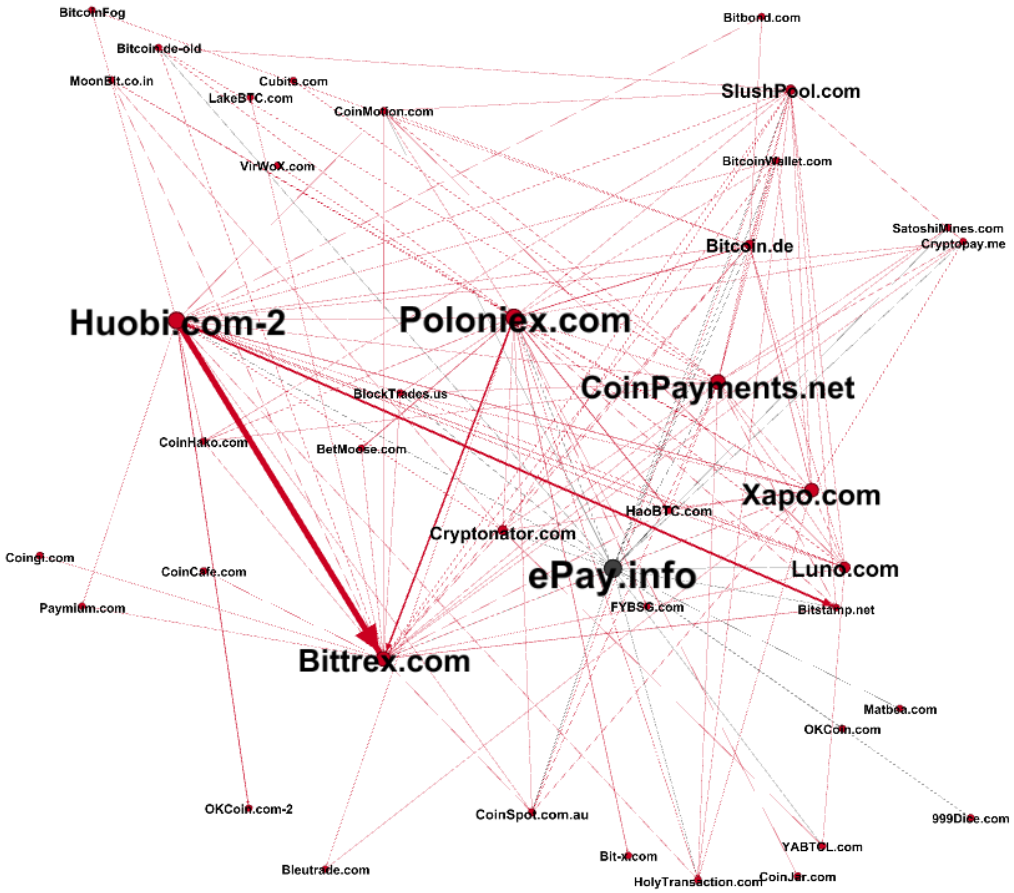
In [30]:
```
#To display the graphs nov20 in Jupyter
Image(filename = "nov20.PNG", width=700, height=100)
```

Out[30]:



In [32]:
```
#To display the graphs dec20 in Jupyter
Image(filename = "dec20.PNG", width=700, height=100)
```

Out[32]:



In [33]:
```
#Create a table with full info on nodes, edges, density, av degree,
var = {'Dec 20': [nx.number_of_nodes(G_dec20), nx.number_of_edges(G
       'Nov 20': [nx.number_of_nodes(G_nov20), nx.number_of_edges(G
       'Feb 19': [nx.number_of_nodes(G_feb19), nx.number_of_edges(G
       'Jan 19': [nx.number_of_nodes(G_jan19), nx.number_of_edges(G
       }
df = pd.DataFrame(var, columns = ['Dec 20', 'Nov 20', 'Feb 19', 'Ja
rounded_df = df.round(decimals=2)
print (rounded_df)
```

	Dec 20	Nov 20	Feb 19	Jan 19
Nodes	40.00	35.00	54.00	52.00
Edges	141.00	159.00	296.00	326.00
Density	0.18	0.27	0.21	0.25
Av. degree	7.05	9.09	10.96	12.54
Transitivity	0.48	0.55	0.49	0.50
Av. clustering	0.57	0.66	0.66	0.74

The table above contains several basic tools to analyze a network. Those tools are presented by line and each column correspond to one period studied:

-The first and the most simple one is the number of nodes. More nodes means more exchange platforms in our network, since we only keep some of their public keys (with letter). We can see it as the size of the network.

-The second one is the number of edges. An edge is the connexion between two nodes. Accordingly, the number of edges can be perceived as a measure of network intensity, although it also depends on the size of edges and their location.

-In third, we can see the density. The density is a good complement of the number of edges to observe the intensity on a network. Indeed, this indicator allows us to observe the connexion between nodes regarding maximum capacity of the network. To be more precise, it is the ratio of pairs of nodes connected by edge in a network. In our case, it is the ratio between the current intensity of bilateral transactions and the maximum intensity of bilteral transactions.

-The forth indicator is the average degree. The degree of a node is the number of its neighbour in the most basic definition. From this point, other elements have been developed as the weighted degree, which correspond to the number of connexions of a node by taking into account the weight of each of those connexions. In this line, we have computed the average basic degree in our entire network, corresponding to the average number of transaction counterparties for each exchange platform. As before, the average degree can be seen as a complement for the study of the intensity and the utilization of the network.

-The fifth indicator is the transitivity also called global clustering coefficient. This is a way to study and modelize the intensity of the network activity through different subgroups of nodes. A triangle is the subgraph built with three nodes connected. The transitivity is computed as the ratio between the existing triangle and the maximum possible number of triangle in our graph. Consequently, we can see this measure of the interconnection between nodes.

-The last indicator is the average clustering coefficient which is another way to observe if there are a lot of cluster possibilities in our graph. This measure is just the arithmetic mean of of existing triangles.

In [36]:
```
###Create a dataframe that shows number of conections of each node
#Creat dictionaries for each year and store there nbr of connection
dict_dec20 = {}
for x in G_dec20.nodes:
 dict_dec20[x] = G_dec20.degree[x]
dict_nov20 = {}
for x in G_nov20.nodes:
 dict_nov20[x] = len(G_nov20[x])
dict_feb19 = {}
for x in G_feb19.nodes:
 dict_feb19[x] = len(G_feb19[x])
dict_jan19 = {}
for x in G_jan19.nodes:
 dict_jan19[x] = len(G_jan19[x])
```

```
#Conver the dictionaries into pandas series
s = pd.Series(dict_dec20, name='Dec 20')
s_1 = pd.Series(dict_nov20, name='Nov 20')
s_2 = pd.Series(dict_feb19, name='Feb 19')
s_3 = pd.Series(dict_jan19, name='Jan 19')
#Make the data frames, sort the dec descending order
df1 = s.to_frame().sort_values('Dec 20', ascending=False)
df2 = s_1.to_frame()
df3 = s_2.to_frame()
df4 = s_3.to_frame()
#Concatinate all dataframes
Connection_table = pd.concat([df1, df2, df3, df4], axis=1)

print(Connection_table)
```

	Dec 20	Nov 20	Feb 19	Jan 19
Poloniex.com	26.0	21.0	34.0	33.0
Bittrex.com	25.0	22.0	35.0	37.0
ePay.info	23.0	24.0	40.0	40.0
Huobi.com-2	22.0	22.0	27.0	29.0
CoinPayments.net	21.0	24.0	32.0	36.0
Xapo.com	17.0	16.0	23.0	26.0
SlushPool.com	16.0	17.0	29.0	30.0
Bitcoin.de	13.0	16.0	18.0	22.0
Luno.com	12.0	16.0	21.0	22.0
Cryptonator.com	10.0	21.0	26.0	30.0
CoinMotion.com	8.0	8.0	9.0	12.0
Cryptopay.me	7.0	8.0	14.0	20.0
HaoBTC.com	7.0	8.0	17.0	16.0
BlockTrades.us	7.0	5.0	7.0	11.0
CoinSpot.com.au	7.0	9.0	11.0	10.0
BitcoinWallet.com	7.0	8.0	11.0	10.0
HolyTransaction.com	6.0	10.0	11.0	14.0
Bitstamp.net	6.0	8.0	38.0	37.0
Bitcoin.de-old	5.0	7.0	3.0	9.0
CoinHako.com	5.0	5.0	10.0	10.0
MoonBit.co.in	4.0	9.0	15.0	16.0
YABTCL.com	3.0	6.0	8.0	9.0
SatoshiMines.com	3.0	5.0	6.0	5.0
Paymium.com	2.0	NaN	9.0	11.0
Bitbond.com	2.0	1.0	8.0	9.0
Bleutrade.com	2.0	2.0	11.0	8.0
BetMoose.com	2.0	4.0	3.0	1.0
BitcoinFog	2.0	3.0	4.0	8.0
FYBSG.com	1.0	NaN	3.0	9.0
LakeBTC.com	1.0	NaN	3.0	4.0
CoinJar.com	1.0	1.0	11.0	15.0
Bit-x.com	1.0	1.0	6.0	7.0
CoinCafe.com	1.0	2.0	1.0	3.0
OKCoin.com	1.0	NaN	1.0	1.0
999Dice.com	1.0	4.0	14.0	14.0
VirWoX.com	1.0	1.0	7.0	5.0
Matbea.com	1.0	1.0	1.0	2.0
Coingi.com	1.0	2.0	1.0	NaN
Cubits.com	1.0	NaN	1.0	NaN
OKCoin.com-2	1.0	NaN	1.0	2.0
BTCCPool	NaN	1.0	NaN	NaN

BitoEX.com	NaN	NaN	10.0	11.0
Vaultoro.com	NaN	NaN	7.0	7.0
BitBargain.co.uk	NaN	NaN	2.0	3.0
SatoshiDice.com	NaN	NaN	5.0	5.0
Hashnest.com	NaN	NaN	10.0	9.0
Bter.com-old	NaN	NaN	1.0	NaN
AnxPro.com	NaN	NaN	6.0	8.0
CoinTrader.net	NaN	NaN	6.0	8.0
SafeDice.com	NaN	NaN	7.0	5.0
HitBtc.com	NaN	NaN	2.0	2.0
SatoshiCircle.com	NaN	NaN	1.0	NaN
HappyCoins.com	NaN	NaN	3.0	5.0
TheRockTrading.com	NaN	NaN	1.0	2.0
LocalBitcoins.com	NaN	NaN	1.0	1.0
CoinGaming.io	NaN	NaN	NaN	2.0
Cex.io	NaN	NaN	NaN	1.0

The table above shows how the degree for each node changes in the bullish and bearish market.

Detecting top nodes in bullish and bearish markets

In [37]:
```
###Top 5 nodes by degree (similar to connection table)
#Copute the degree
degree_dict = dict(G_dec20.degree(G_dec20.nodes()))
#Sort it
sorted_degree = sorted(degree_dict.items(), key=itemgetter(1), reve
#Print
print("Top 5 nodes in bullish market (dec 20) by degree:")
for a in sorted_degree[:5]:
    print(a)

print (' ')

degree_dict1 = dict(G_feb19.degree(G_feb19.nodes()))
sorted_degree1 = sorted(degree_dict1.items(), key=itemgetter(1), re
print("Top 5 nodes in bearish market (feb 19) by degree:")
for b in sorted_degree1[:5]:
    print( b)
```

```
Top 5 nodes in bullish market (dec 20) by degree:
('Poloniex.com', 26)
('Bittrex.com', 25)
('ePay.info', 23)
('Huobi.com-2', 22)
('CoinPayments.net', 21)

Top 5 nodes in bearish market (feb 19) by degree:
('ePay.info', 40)
('Bitstamp.net', 38)
('Bittrex.com', 35)
('Poloniex.com', 34)
('CoinPayments.net', 32)
```

In [38]:
```
###Top 5 nodes by betweeness centrality
#Copute bet centrality for all nodes
betweenness_dict = nx.betweenness_centrality(G_dec20) # Run betweenn
#Sort it in ascending order
sorted_betweenness = sorted(betweenness_dict.items(), key=itemgette
#Print top 5
print("Top 5 nodes in bullish market (dec 20) by betweenness centra
for c in sorted_betweenness[:5]:
    print(c)

print (' ') #Space

betweenness_dict1 = nx.betweenness_centrality(G_feb19) # Run betwee
sorted_betweenness1 = sorted(betweenness_dict1.items(), key=itemget
print("Top 5 nodes in bearish market (feb 19) by betweenness centra
for d in sorted_betweenness1[:5]:
    print(d)
```

```
Top 5 nodes in bullish market (dec 20) by betweenness centrality:
('Poloniex.com', 0.2740060621639569)
('Bittrex.com', 0.23172086198401987)
('ePay.info', 0.1993097058886533)
('Huobi.com-2', 0.1462732686416897)
('CoinPayments.net', 0.08529711029711029)

Top 5 nodes in bearish market (feb 19) by betweenness centrality:
('ePay.info', 0.2326098641319977)
('Bitstamp.net', 0.16779818210333453)
('SlushPool.com', 0.1237560999716297)
('Bittrex.com', 0.11642738309102604)
('Poloniex.com', 0.08616164516019378)
```

As the name suggests, centrality measures allow us to observe if some subgraphs or nodes are more central in the network. However there exists several definitions of centrality. For example it can mean a geographical centrality or a centrality linked with the node size. In this work, we chose the betweenness centrality measure: in this case centrality is observed from another point of view which is the importance of some nodes in terms of traffic. Accordingly, a central node is seen as a node which is a bridge, a crossing point for other nodes.

In [40]:
```
#Top 5 nodes by PageRank
pagerank_dict = nx.pagerank(G_dec20)
sorted_pagerank = sorted(pagerank_dict.items(), key=itemgetter(1),
print("Top 5 nodes in bullish market (dec 20) by PageRank:")
for b in sorted_pagerank [:5]:
    print(b)

print (' ')

pagerank_dict1 = nx.pagerank(G_feb19)
sorted_pagerank1 = sorted(pagerank_dict1.items(), key=itemgetter(1)
print("Top 5 nodes in bearish market (feb 19) by PageRank:")
for b in sorted_pagerank1 [:5]:
    print(b)
```

```
Top 5 nodes in bullish market (dec 20) by PageRank:
('Poloniex.com', 0.091924882198585)
('Bittrex.com', 0.08671344900277465)
('ePay.info', 0.07857240207095831)
('Huobi.com-2', 0.07228524350313963)
('CoinPayments.net', 0.06688063322459913)

Top 5 nodes in bearish market (feb 19) by PageRank:
('ePay.info', 0.06963703838744668)
('Bitstamp.net', 0.062161345688136964)
('Bittrex.com', 0.05534181685489479)
('Poloniex.com', 0.052788321141497 46)
('SlushPool.com', 0.049325203681437765)
```

PageRank computes a ranking of the nodes in the graph G based on the structure of the incoming links. It was originally designed as an algorithm to rank web pages.

Conclusion

Finally, our objective is to analyse the exchange network between bitcoin centralized exchange platforms and to compare it in different time periods. Periods studied was February 2019 and January 2019 (bearish market for Bitcoin) and November and December 2020 (bullish market for Bitcoin). To analyse networks of each period, we used the package "networkx" on Python for numerical describing indicators and Gephi for visual observation of graph. As numerical indicators, on the one hand we computed the number of nodes, the number of edges, the density, the average degree and the transitivity for network intensity measures and on the other hand we computed average clustering coefficient and betweenness centrality to measure the grouping of nodes and the centralization in this network.

Since the 2019 was a bearish market and 2020 a bullish one, we expected the 2020 networks to have more nodes, edges and a stronger density and average degree than 2019 ones. Indeed, the bullish market is traditionally characterized as more liquid and fluent than a bearish one. Moreover, it would be normal to think that the interest for bitcoin increased with the bullish market meaning more intervenant on the market and therefore more exchanges (nodes).

Concerning the top five members of bullish and bearish market, we can see that Poloniex has the highest centrality and the highest PageRank in bullish market (dec 2020) which makes it the most important node. For the bearish market, ePay is the most important node.

However the results obtained do not correspond to the hypotheses formulated. Indeed, we can see that over November 2020 and December 2020 the number of nodes and edges are less important. They are also of a lower intensity. We also observe results that are not in line with our hypotheses for the average degree and transitivity.

These unexpected results are most probably explained by the fact that our lexicon of public keys is not accurate and updated for the end of 2020. Centralised exchanges for confidentiality reasons and to restructure their technical infrastructure are likely to replace their public key pool in order to better manage and optimise internal flows.

The results obtained over the period Jan 2019 - February 2019 would therefore be rather representative of the real flows observed, while those over the period Nov 2020 - Dec 2020 would be much less representative. It is therefore quite possible to affirm that the public key lexicon on which our work is based is at the origin of the inconsistency of our results.