EECS E6893 Big Data Analytics - Homework Assignment 4

Name: Qi Wang

UNI: qw2261

Base

In [10]:
```
from pyspark import SparkConf, SparkContext
import pyspark
import sys
from collections import defaultdict
import pandas as pd
```

In [11]:
```
# Configure Spark
sc = pyspark.SparkContext.getOrCreate()
# The directory for the file
filename = "gs://homework0_qi/HW2/q1.txt"
```

In [12]:
```
def getData(sc, filename):
    """
    Load data from raw text file into RDD and transform.
    Hint: transfromation you will use: map(<lambda function>).
    Args:
        sc (SparkContext): spark context.
        filename (string): hw2.txt cloud storage URI.
    Returns:
        RDD: RDD list of tuple of (<User>, [friend1, friend2, ... ]),
        each user and a list of user's friends
    """
    # read text file into RDD
    data = sc.textFile(filename)

    # TODO: implement your logic here
    data = data.map(lambda line: line.split("\t"))
    data = data.map(lambda line: (int(line[0]), [int(each) for each in line[1].split(',')] if len(line[1]) else

    return data
```

In [22]:
```
sc.stop()
```

Graph Analysis

In [14]:
```
# Install graphframes
!pip install "git+https://github.com/munro/graphframes.git@release-0.5.0#egg=graphframes&subdirectory=python"
```

```
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Please upgrade your Python as Pyth
on 2.7 won't be maintained after that date. A future version of pip will drop support for Python 2.7. More deta
ils about Python 2 support in pip, can be found at https://pip.pypa.io/en/latest/development/release-process/#p
ython-2-support (https://pip.pypa.io/en/latest/development/release-process/#python-2-support)
Requirement already satisfied: graphframes from git+https://github.com/munro/graphframes.git@release-0.5.0#egg=
graphframes&subdirectory=python in /opt/conda/anaconda/lib/python2.7/site-packages (0.5.0)
```

In [20]:
```
from graphframes import *
from pyspark import SQLContext
import os
```

In [23]:
```
# Configure Spark
if not os.path.isdir("checkpoints"):
    os.mkdir("checkpoints")
conf = SparkConf().setMaster("local").setAppName('connected components')
sc = SparkContext(conf = conf)
```

In [24]:
```
# Configure sqlcontext and directory
sqlContext = SQLContext(sc)
SparkContext.setCheckpointDir(sc, "checkpoints")
```

In [25]:
```
# Get data in proper format
data = getData(sc, filename)
```

In [26]:
```
def getVertices(data, sqlContext):
    """
    Get the vertices of the friends network

    Args:
        data: RDD: RDD list of tuple of (<User>, [friend1, friend2, ... ]), each user and a list of user's frienc
        sqlContext: SQLContext

    Returns:
        vertice: ID DataFrame of all users
    """

    return sqlContext.createDataFrame(data.map(lambda line: (str(line[0]), )), schema = ["id"])
```

In [27]:
```
vertices = getVertices(data, sqlContext)
```

In [28]:
```
vertices.show()
```

```
+---+
| id|
+---+
|  0|
|  1|
|  2|
|  3|
|  4|
|  5|
|  6|
|  7|
|  8|
|  9|
| 10|
| 11|
| 12|
| 13|
| 14|
| 15|
| 16|
| 17|
| 18|
| 19|
+---+
only showing top 20 rows
```

In [29]:
```
def friendRelationBuid(line):
    '''
    Get the egde of the friends network

    Args:
        line (tuple): a tuple of (<User1>, [(<User2>, 0), (<User3>, 1)....])

    Yields:
        friend relation (tuple): a tuple of (friend1, friend2)
    '''
    user = line[0]
    friends = line[1]

    for each_friend in friends:
        yield (user, each_friend)
```

In [30]:
```
def getEdges(data, sqlContext):
    edges = data.flatMap(friendRelationBuid)
    return sqlContext.createDataFrame(edges, schema = ["src", "dst"])
```

In [31]:
```
edges = getEdges(data, sqlContext)
```

In [32]:
```
edges.show()
```

```
+---+---+
|src|dst|
+---+---+
|  0|  1|
|  0|  2|
|  0|  3|
|  0|  4|
|  0|  5|
|  0|  6|
|  0|  7|
|  0|  8|
|  0|  9|
|  0| 10|
|  0| 11|
|  0| 12|
|  0| 13|
|  0| 14|
|  0| 15|
|  0| 16|
|  0| 17|
|  0| 18|
|  0| 19|
|  0| 20|
+---+---+
only showing top 20 rows
```

In [33]:
```
# Build graph
graph = GraphFrame(vertices, edges)
```

In [34]:
```
result = graph.connectedComponents()
```

In [35]:
```
result.show()
```

```
+---+---------+
| id|component|
+---+---------+
|  0|        0|
|  1|        0|
|  2|        0|
|  3|        0|
|  4|        0|
|  5|        0|
|  6|        0|
|  7|        0|
|  8|        0|
|  9|        0|
| 10|        0|
| 11|        0|
| 12|        0|
| 13|        0|
| 14|        0|
| 15|        0|
| 16|        0|
| 17|        0|
| 18|        0|
| 19|        0|
+---+---------+
only showing top 20 rows
```

Extract node and linked from 103079215141 component

In [36]:
```
nodes_req = result.filter(result['component'] == '103079215141').collect()
```

In [37]:
```
nodes = []
```

In [38]:
```
for each in nodes_req:
    nodes.append(each['id'])
```

In [39]:
```
nodes
```

Out[39]:
```
[u'18233',
 u'18234',
 u'18235',
 u'18236',
 u'18237',
 u'18238',
 u'18239',
 u'18240',
 u'18241',
 u'18242',
 u'18243',
 u'18244',
 u'18245',
 u'18246',
 u'18247',
 u'18248',
 u'18249',
 u'18250',
 u'18251',
 u'18252',
 u'18253',
 u'18254',
 u'18255',
 u'18256',
 u'18257']
```

In [40]:
```
df_node = pd.DataFrame(nodes, index = range(1, len(nodes) + 1), columns = ['node'])
df_node
```

Out[40]:
```
     node
 1  18233
 2  18234
 3  18235
 4  18236
 5  18237
 6  18238
 7  18239
 8  18240
 9  18241
10  18242
11  18243
12  18244
13  18245
14  18246
15  18247
16  18248
17  18249
18  18250
19  18251
20  18252
21  18253
22  18254
23  18255
24  18256
```

	node
25	18257

In [41]:
```
df_node.to_csv("node.csv")
!gsutil cp 'node.csv' 'gs://big_data_hw4'
```

```
Copying file://node.csv [Content-Type=text/csv]...
/ [1 files][  222.0 B/  222.0 B]
Operation completed over 1 objects/222.0 B.
```

In [42]:
```
linked = []
```

In [43]:
```
for each in nodes:
    linked.extend(edges.filter(edges['src'] == each).collect())
```

In [44]:
```
linked
```

Out[44]:
```
[Row(src=18233, dst=18234),
 Row(src=18233, dst=18235),
 Row(src=18233, dst=18236),
 Row(src=18233, dst=18237),
 Row(src=18233, dst=18238),
 Row(src=18233, dst=18239),
 Row(src=18233, dst=18240),
 Row(src=18233, dst=18241),
 Row(src=18233, dst=18242),
 Row(src=18233, dst=18243),
 Row(src=18233, dst=18244),
 Row(src=18233, dst=18245),
 Row(src=18233, dst=18246),
 Row(src=18233, dst=18247),
 Row(src=18233, dst=18248),
 Row(src=18233, dst=18249),
 Row(src=18233, dst=18250),
 Row(src=18233, dst=18251),
 Row(src=18233, dst=18252),
```

In [45]:
```
src_dst = []
```

In [46]:
```
for each in linked:
    src_dst.append([int(each['src']) - 18233, int(each['dst']) - 18233])
```

In [47]:
```
df_linked = pd.DataFrame(src_dst, index = range(1, len(src_dst) + 1), columns = ['source', 'target'])
df_linked
```

Out[47]:

	source	target
1	0	1
2	0	2
3	0	3
4	0	4
5	0	5
6	0	6
7	0	7
8	0	8
9	0	9
10	0	10
11	0	11
12	0	12
13	0	13
14	0	14
15	0	15
16	0	16
17	0	17
18	0	18
19	0	19
20	0	20
21	0	21
22	0	22
23	0	23
24	0	24

	source	target
25	1	0
26	1	2
27	1	4
28	1	5
29	1	6
30	1	7
...	...	...
423	22	2
424	22	5
425	22	13
426	22	17
427	22	19
428	22	24
429	23	0
430	23	6
431	23	12
432	24	0
433	24	1
434	24	2
435	24	4
436	24	6
437	24	7
438	24	8
439	24	9
440	24	10
441	24	11
442	24	12

	source	target
443	24	13
444	24	15
445	24	16
446	24	17
447	24	19
448	24	20
449	24	22
450	24	5
451	24	14
452	24	18

452 rows × 2 columns

In [48]:
```
df_linked.to_csv("linked.csv")
!gsutil cp 'linked.csv' 'gs://big_data_hw4'
```

```
Copying file://linked.csv [Content-Type=text/csv]...
/ [1 files][  3.9 KiB/  3.9 KiB]
Operation completed over 1 objects/3.9 KiB.
```

In [ ]: