Big Data Analytics – CS7070

Programming Project #2

Phase 3

Submitted By: Ankit Pandey

M13435273

PySpark Code:

```
%spark.pyspark

# read input text file to RDD

rdd = sc.textFile("/tmp/data/tinyDataset.txt")

#Define the List with the list constructor

list_rdd=list()

#Store the rdd in List with the help of collect

list_rdd=rdd.collect()

#Iterating through the Loop to display the graph taken as Input

print("TinyDataSet Graph: (List of edges) as Input:")

for x in range(len(list_rdd)):

   print(list_rdd[x])

#Splitting the RDD at spaces

rdd2=rdd.map(lambda x: x.split())

#FlatMap will flatten multiple list into single list and storing the verices of the graph in two way form assuming it to be a undirected graph

rdd3=rdd2.flatMap(lambda y:[[y[0],y[1]],[y[1],y[0]]])

# reduceByKey Merges the values for each key. It will perform the merging locally on each mapper before sending results to a reducer, similarly to a “combiner” in MapReduce.

rdd4=rdd3.reduceByKey(lambda k,v:k+","+v)

#Mapped to again get the nodes in List format for 2- hop computation

rdd5=rdd4.map(lambda z :[z[0], z[1].split(",")])

#Starting 2 Hop Projection of Graph
```

```
rdd6=rdd5.flatMap(lambda x : [[(x[0],x[1][l]),x[1]] for l in range(len(x[1]))])

rdd7=rdd5.flatMap(lambda x : [[(x[1][m],x[0]),x[1]] for m in range(len(x[1]))])

rdd8 = rdd6.join(rdd7)

rdd9 = rdd8.map(lambda x: [x[0][0] , [x[0][1],x[1][1]]])

rdd10=rdd9.reduceByKey(lambda k,v : k + v)
```

#Phase3 -Triangles from 2 Hop Projection

```
rdd11=rdd10. flatMap(lambda x : [[(x[0],x[1][k][0],x[1][k][1][m]),1] for k in range(len(x[1])) for m in range(len(x[1][k][1]))])

rdd12=rdd11.join(map4).reduceByKey(lambda x,y: x+y)

rdd13=rdd11.join(rdd12)

rdd14=rdd13.reduceByKey(lambda x,y: x+y)

rdd15=rdd14.map(lambda x : ((x[0]),x[0][0])).reduceByKey(lambda x,y: x)

#print(rdd15.collect())

l=rdd15.collect()

#set1=set()

#tup=tuple()

#for i in range(len(l)):

#    set1.add(l[i][0])

#print(set1)

unique_data=[list(i) for i in set(tuple(i) for i in l)]

tup=tuple()

for x in range(len(unique_data)) :

   tup=x[i]

   print(tup)
```

Output 1:

(8, [(5, 8, 9)])

(2, [(2, 3, 6)])

(4, [(4, 5, 6), (1, 3, 4), (3, 4, 6)])

(6, [(2, 3, 6), (3, 4, 6), (4, 5, 6)])

(1, [(1, 3, 4)])

(9, [(5, 8, 9)])

(3, [(1, 3, 4), (3, 4, 6), (2, 3, 6)])

(5, [(5, 8, 9), (4, 5, 6)])

Output2:

(8, [(7, 8, 12), (8, 9, 13), (4, 8, 9), (4, 8, 13), (8, 12, 13), (3, 4, 8), (3, 8, 12), (3, 7, 8)])

(16, [(16, 17, 21), (12, 16, 17), (16, 20, 21), (11, 12, 16), (11, 15, 16)])

(32, [(28, 30, 32), (28, 31, 32), (30, 31, 32), (27, 31, 32), (23, 27, 32)])

(24, [(20, 24, 29), (24, 28, 29), (19, 24, 28), (19, 20, 24)])

(3, [(3, 4, 8), (3, 8, 12), (3, 7, 12), (2, 3, 7), (1, 2, 3), (3, 4, 5), (3, 7, 8)])

(19, [(19, 20, 24), (19, 24, 28), (10, 15, 19)])

(11, [(11, 12, 16), (7, 11, 12), (2, 7, 11), (10, 11, 15), (6, 7, 11), (2, 6, 11), (6, 10, 11), (11, 15, 16)])

(27, [(26, 27, 31), (22, 23, 27), (22, 27, 31), (22, 26, 27), (27, 31, 32), (23, 27, 32)])

(28, [(28, 29, 31), (28, 29, 30), (28, 30, 32), (19, 24, 28), (28, 30, 31), (24, 28, 29), (28, 31, 32)])

(12, [(12, 16, 17), (7, 8, 12), (3, 7, 12), (7, 11, 12), (3, 8, 12), (11, 12, 16), (12, 13, 17), (8, 12, 13)])

(4, [(4, 8, 9), (3, 4, 8), (3, 4, 5), (4, 5, 9), (4, 9, 13), (4, 8, 13)])

(20, [(19, 20, 24), (20, 25, 29), (20, 21, 25), (20, 24, 29), (16, 20, 21)])

(7, [(7, 8, 12), (3, 7, 8), (6, 7, 11), (2, 3, 7), (2, 7, 11), (2, 6, 7), (7, 11, 12), (3, 7, 12)])

(31, [(27, 31, 32), (29, 30, 31), (28, 29, 31), (22, 27, 31), (28, 31, 32), (28, 30, 31), (26, 27, 31), (30, 31, 32), (26, 30, 31), (22, 26, 31)])

(15, [(11, 15, 16), (10, 11, 15), (10, 15, 19)])

(23, [(23, 27, 32), (22, 23, 27), (14, 18, 23), (18, 22, 23)])

(9, [(4, 5, 9), (4, 9, 13), (4, 8, 9), (9, 13, 14), (8, 9, 13)])

(25, [(20, 25, 29), (20, 21, 25), (25, 29, 30), (21, 25, 30)])

(17, [(12, 13, 17), (12, 16, 17), (17, 21, 22), (16, 17, 21)])

(1, [(1, 2, 6), (1, 6, 10), (1, 2, 3)])

(2, [(2, 6, 7), (1, 2, 6), (1, 2, 3), (2, 3, 7), (2, 7, 11), (2, 6, 11)])

(18, [(14, 18, 23), (18, 22, 23), (13, 14, 18)])

(26, [(22, 26, 31), (26, 30, 31), (21, 26, 30), (21, 22, 26), (26, 27, 31), (22, 26, 27)])

(10, [(1, 6, 10), (10, 11, 15), (10, 15, 19), (6, 10, 11)])

(13, [(8, 9, 13), (4, 8, 13), (8, 12, 13), (13, 14, 18), (9, 13, 14), (4, 9, 13), (12, 13, 17)])

(21, [(16, 17, 21), (16, 20, 21), (21, 26, 30), (21, 22, 26), (21, 25, 30), (17, 21, 22), (20, 21, 25)])

(29, [(20, 24, 29), (24, 28, 29), (28, 29, 30), (28, 29, 31), (29, 30, 31), (25, 29, 30), (20, 25, 29)])

(5, [(4, 5, 9), (3, 4, 5)])

(30, [(28, 30, 31), (21, 26, 30), (29, 30, 31), (25, 29, 30), (21, 25, 30), (28, 30, 32), (26, 30, 31), (30, 31, 32), (28, 29, 30)])

(22, [(21, 22, 26), (17, 21, 22), (22, 27, 31), (22, 23, 27), (22, 26, 31), (18, 22, 23), (22, 26, 27)])

(14, [(13, 14, 18), (9, 13, 14), (14, 18, 23)])

(6, [(6, 7, 11), (2, 6, 7), (6, 10, 11), (2, 6, 11), (1, 2, 6), (1, 6, 10)])