Social Media Workshop

Assignment-04

NetworkX Lab

Practice following python code using above mentioned packages and check the results of programs.

1. Practice Code: Giant Component

```
try
   import matplotlib.pyplot as plt
except:
   raise

import networkx as nx
import math

try:
   import pygraphviz
   from networkx.drawing.nx_agraph import graphviz_layout
   layout = graphviz_layout
except ImportError:
   try:
      import pydotplus
      from networkx.drawing.nx_pydot import graphviz_layout
      layout = graphviz_layout
   except ImportError:
      print("PyGraphviz and PyDotPlus not found;\n"
           "drawing with spring layout;\n"
           "will be slow.")
      layout = nx.spring_layout

n=150  # 150 nodes
# p value at which giant component (of size log(n) nodes) is expected
p_giant=1.0/(n-1)
# p value at which graph is expected to become completely connected
p_conn=math.log(n)/float(n)

# the following range of p values should be close to the threshold
pvals=[0.003, 0.006, 0.008, 0.015]

region=220 # for pylab 2x2 subplot layout
plt.subplots_adjust(left=0,right=1,bottom=0,top=0.95,wspace=0.01,hspace=0.01)
for p in pvals:
   G=nx.binomial_graph(n,p)
   pos=layout(G)
   region+=1
   plt.subplot(region)
   plt.title("p = %6.3f"%(p))
   nx.draw(G,pos,
         with_labels=False,
         node_size=10
         )
   # identify largest connected component
   Gcc=sorted(nx.connected_component_subgraphs(G), key = len, reverse=True)
   G0=Gcc[0]
   nx.draw_networkx_edges(G0,pos,
                    with_labels=False,
                    edge_color='r',
                    width=6.0
                  )
   # show other connected components
   for Gi in Gcc[1:]:
     if len(Gi)>1:
       nx.draw_networkx_edges(Gi,pos,
                       with_labels=False,
                       edge_color='r',
                       alpha=0.3,
                            width=5.0
                            )
plt.savefig("giant_component.png")
plt.show() # display
```

Exercise 1

Write Python code using NetworkX package to draw a graph and write python code for the following:

1. How many nodes and edges are in the graph?
2. Write a function *average_degree* to compute the average degree of a networkx graph G.
3. The diameter of a connected component of a graph is the longest shortest path in the graph, use networkx function diameter to computes diameter.
4. Networkx gives you the ability to construct subgraphs. Networkx function subgraph takes two inputs: a network and a subset of the nodes. Write python code for the following:
    a. How many connected components does the subgraph have?
    b. What is the size of the largest component?
    c. Draw the largest component and save the figure as “largest_connected_component.png”.

(Note: Use help (nx.subgraph) to find how to construct the subgraph)

5. Write a function called make_largest_diameter_graph which takes an integer N as input and returns an undirected networkx graph with N nodes that has the largest possible diameter. Print the graph’s average degree and its centers for N = 10. Draw the graph for N = 10 and save it as “largest_diameter_10.png”.
6. Write a function called make_smallest_diameter_graph which takes an integer N as input and returns an undirected networkx graph with N nodes that has the smallest possible diameter. Print the graph’s average degree and its centers for N = 10. Draw the graph for N = 10 and save it as “smallest_diameter_10.png”.
7. Write a function called make_circle_graph which takes an integer N as input and returns an undirected networkx graph with N nodes in a circle. Print the graph’s average degree and its centers for N = 10. Draw the graph for N = 10 and save it as “circle_10.png”.
8. Write a function called make_complete_graph which takes an integer N and returns an undirected networkx graph with N nodes where each node is connected to every other node in the graph. Print the graph’s centers for N = 10. Do not use the networkx built-in function to make a complete graph! Draw the graph for N = 10 and save it as “complete_graph_10.png”.