HOMEWORK OF NETWORKING FOR BIG DATA

29/5/2019

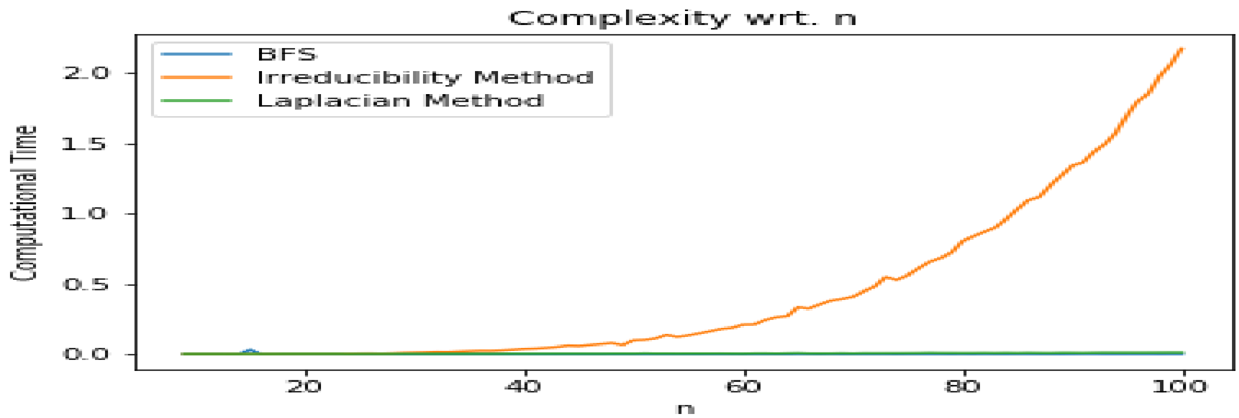
THEODOROS SOFIANOS

Numero di matricola: **1867968**

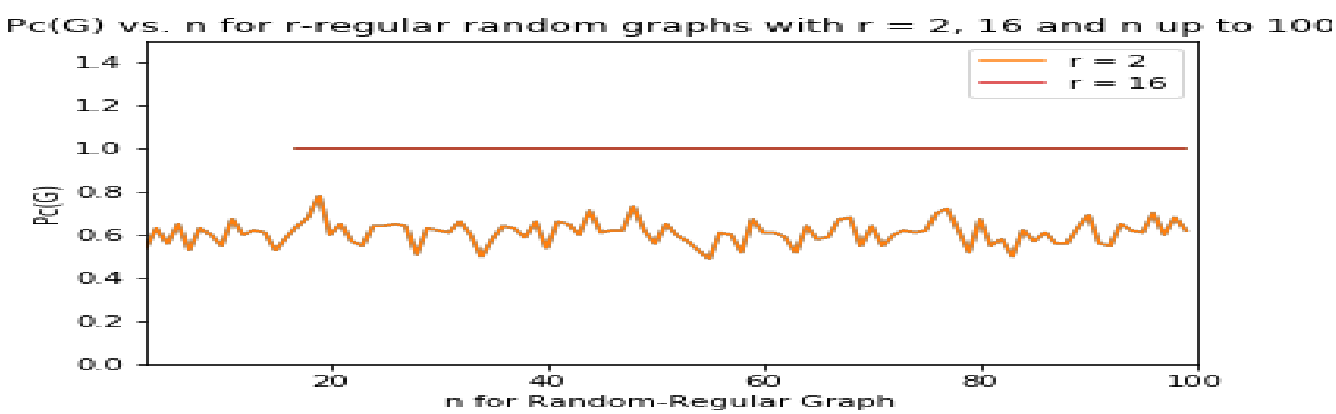
Pseudo-code for creating a regular random graph

```
create regular graph (nodes, neighbours ) # name of the function and inputs

# give as input the number of nodes and neighbours of each node

If nodes> neighbours and nodes*neighbours is even number

# the conditions in which a random regular graph can be created


    create graph # empty graph

    graph.add_nodes for i in  nodes  # add the number of nodes that we want

     for i in graph.nodes

          for j in graph.nodes

               if degree(i)< neighbours

                # if the node i still doesn't have the required number of neighbours


                         sort nodes by degree, ascending order    # the algorithm will try to
    create edge between i and the node of the graph with the lowest degree



                        if the  node with lowest degree != i

    # if the node with the lowest degree in the graph is not i, because we don't want the
    node to create edge with itself

                                  graph.add_edge(i,node with lowest degree)
```

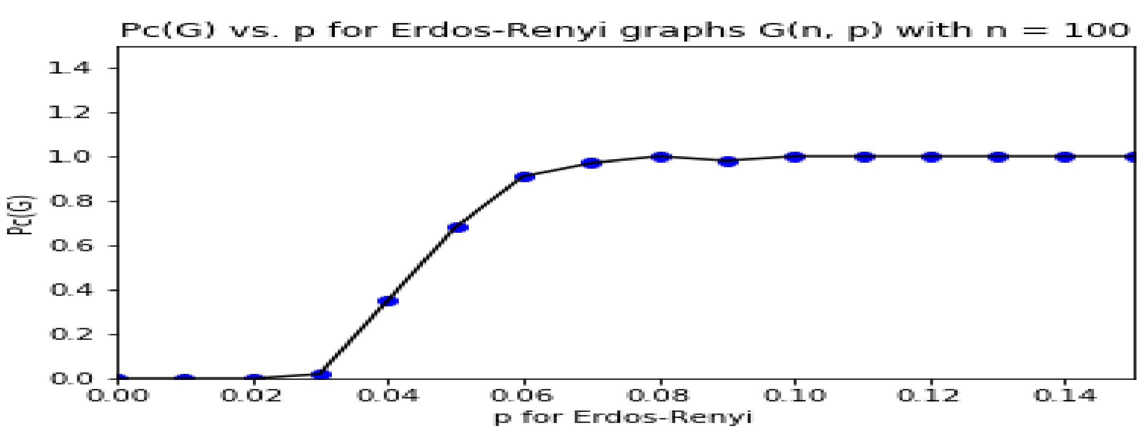
Connectivity check graph. The line of Laplacian method coincides with the one of the BFS. It's clear that BFS or Laplacian are the best methods to check if a graph is connected, since they run in constant time.

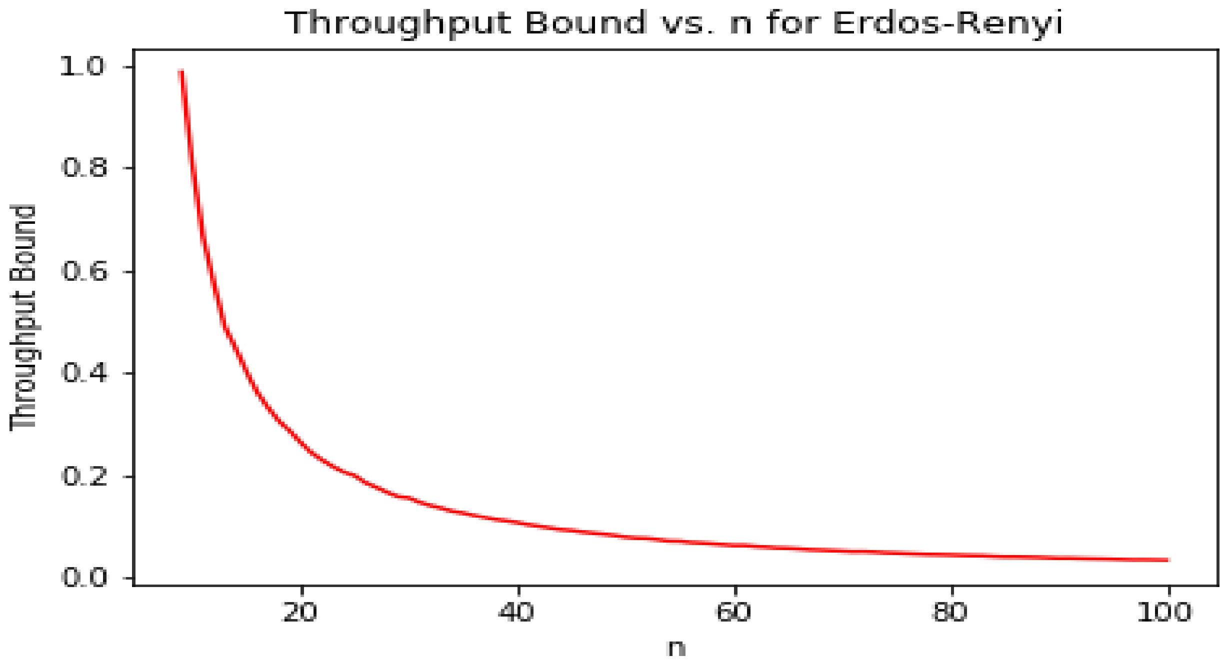


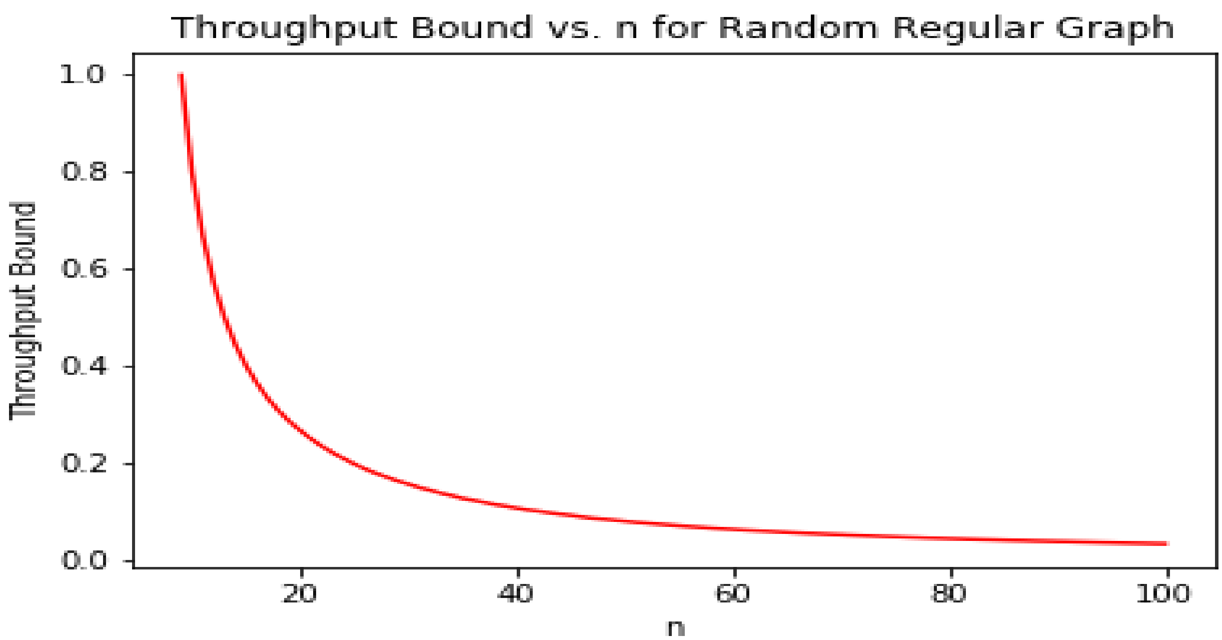
We notice that for r=16, the regular graph will always be connected , for nodes>17. For r=2, the probability of the graph being connected is between 0.5 and 0.8 and does not increase with n.



For p of Erdos-Renyi above 0.07, the graph will almost certainly be connected. For less than 0.03, certainly not. Biggest slope is noticed going from p=0.03 to 0.04(from 0 to 40%).







We can conclude that the throughput bound with respect to n for Erdos-Renyi and Regular Graph for p= 8/(n-1) and r = 8 respectively, is the same, since the 2 curves have the same slope and coincide perfectly. This makes perfect sense, since r=8 means that every node of the graph has 8 neighbouring nodes, and p=8/(n-1) is the probability of a node having another node as neighbor. In an Erdos-Renyi graph with p= 8/(n-1), we expect to have p*(n-1) neighbours, which is equal to r.