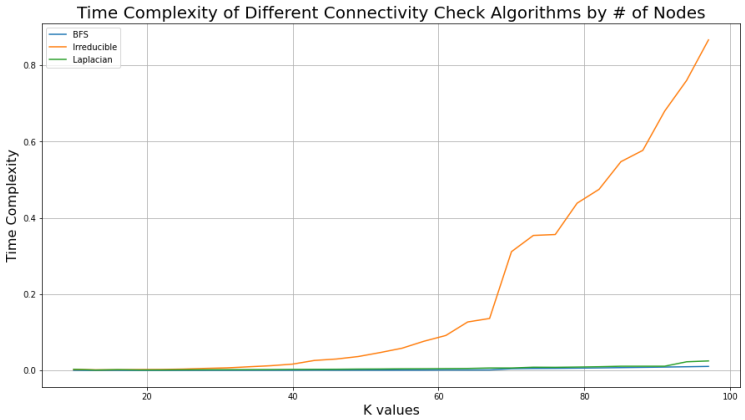
Mert Yildiz yildiz.1951070@studenti.uniroma1.it
Gaurav Ramse ramse.1965564@studenti.uniroma1.it
Tran Luong Bang tran.1956419@studenti.uniroma1.it

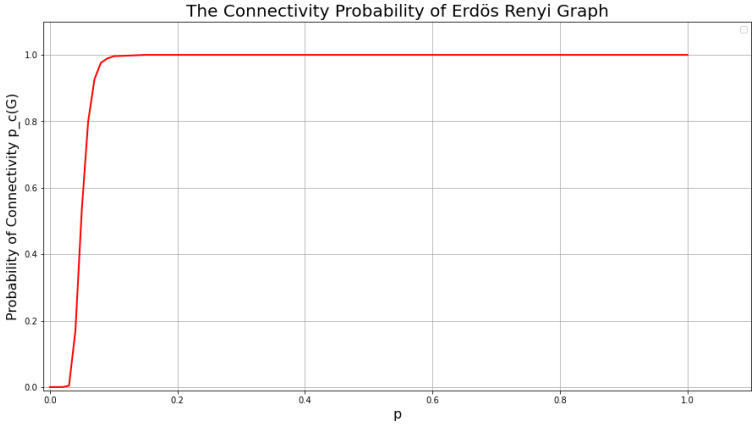
Part 1

We have started to create Erdös Renyi graphs and r-Regular graphs. Then we have written down the codes for the algorithms to calculate the connectivity of the graphs.

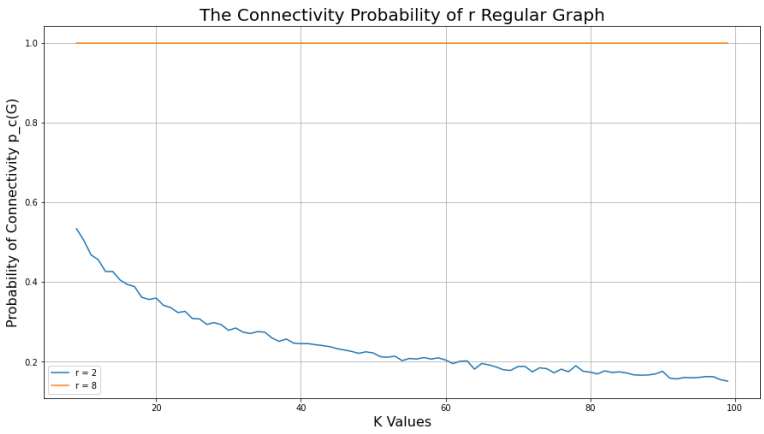
i-) In this part, we have calculated the time complexity for different connectivity check algorithms. As it is shown on the right plot we can say that the Breadth First Search and Laplacian connectivity check algorithms' time complexity does not depend on the number of nodes K while the irreducibility connectivity check algorithm's time complexity increases dramatically with the number of nodes K.



ii-) In this part, we have found the probability of connectivity of the **erdos_renyi_graph**(K, p) with number of nodes K = 100 and different probability p. We have found the connectivity probability with Monte Carlo simulations by generating the graphs 5000 times and checked the connectivity probability. As it can be seen in the **"The Connectivity Probability of Erdös Renyi Graph"** the probability hits 1 just after $p \sim 0.01$.



iii-) In this part, we have found the probability of connectivity of the **random_regular_graph**(r, K) with K = [9,100] and r= 2 and r= 8. We generated the graphs for K > 8 to satisfy r < K condition. We have found the connectivity probability with Monte Carlo simulations by generating the graphs 5000 times for each r and checked the connectivity probability. As it can be seen in the **"The Connectivity Probability of r Regular Graph"**, the graphs are connected with probability $p_c(G) = 1$ for r=8 but the connectivity probability decreases while the number of nodes K increases for r= 2 and it makes sense to not have a connected graph with 2 links for each node.



Code link: https://git.io/JOw5M

Mert Yildiz yildiz.1951070@studenti.uniroma1.it
Gaurav Ramse ramse.1965564@studenti.uniroma1.it
Tran Luong Bang tran.1956419@studenti.uniroma1.it

Part 2

i-) In this part, we have derived the number of switch ports r to be connected to other switches in Jellyfish as a function of n with the same values of N,S and L for the Fat Tree.

	Fat Tree	**Jellyfish**
S # of switches	$\frac{5n^2}{4}$	S
N # of server	$\frac{n^3}{4}$	$S(n-r)$
L # of links connecting switches	$\frac{n^3}{2}$	$\frac{Sr}{2}$

After combining the formulations for Fat Tree and Jellyfish we have derived the the number of switch ports r as below:

$$\boldsymbol{r = \frac{4n}{5}}$$

ii-) In this section, we have derived the expression of the application-oblivious throughput bound TH for an all-to-all traffic matrix among servers as a function of $\underline{h}$ and n only. Since, we had to find the TH for all-to-all traffic we derived the formula of v_f as combination $C(N,2)$ to consider all shortest path pairs between servers. After all simplifications we found the TH bound formular as shown below.

$l = \frac{3n^3}{4}$ total number of links

$v_f = \frac{N(N-1)}{2} = \frac{n^6 - 4n^3}{32}$ number of traffic flows

$\overline{h}$ mean shortest path lengths for server-to-server paths.

$$\boldsymbol{TH \leq \frac{l}{\overline{h}\ v_f}}$$

As a result:

$$\boldsymbol{TH \leq \frac{24}{\overline{h}\,(n^3 - 4)}}$$

iii-) In this part, we have calculated the $\boldsymbol{TH}$ for n = 20, 30, 40, 50, 60

Using the exact value of h as a function of n for Fat-Tree (you must calculate it!) and the lower bound of h for r-regular random graphs (see slides, but be careful with notation) for Jellyfish, evaluate TH for n = 20, 30, 40, 50, 60.

n	**N**	**S**	**L**	**TH_Fat_Tree**	**TH_Jellyfish**
20	2000	500	4000	0.000509	0.001223
30	6750	1125	13500	0.00015	0.000360
40	16000	2000	32000	0.000063	0.000152
50	31250	3125	62500	0.000032	0.000078
60	54000	4500	108000	0.000019	0.000045

Code link: https://git.io/JOw5M