NETWORK SCIENCE OF ONLINE INTERACTIONS

Chapter 5 exercises

Joao Neto

25/May/2023

CHAPTER 5 EXERCISES

- Exercises 5.6, 5.10, 5.15, 5.21

5.6 Consider the family of random networks with average degree $\langle k \rangle = 10$. How many nodes are we likely to need in order to generate such a network with average path length $\langle \ell \rangle = 3.0$? (*Hint*: If you use a guess and check strategy, make sure that $\langle k \rangle$ stays the same for the various values of N. Each will involve a different value of p.)

a. 60

b. 100

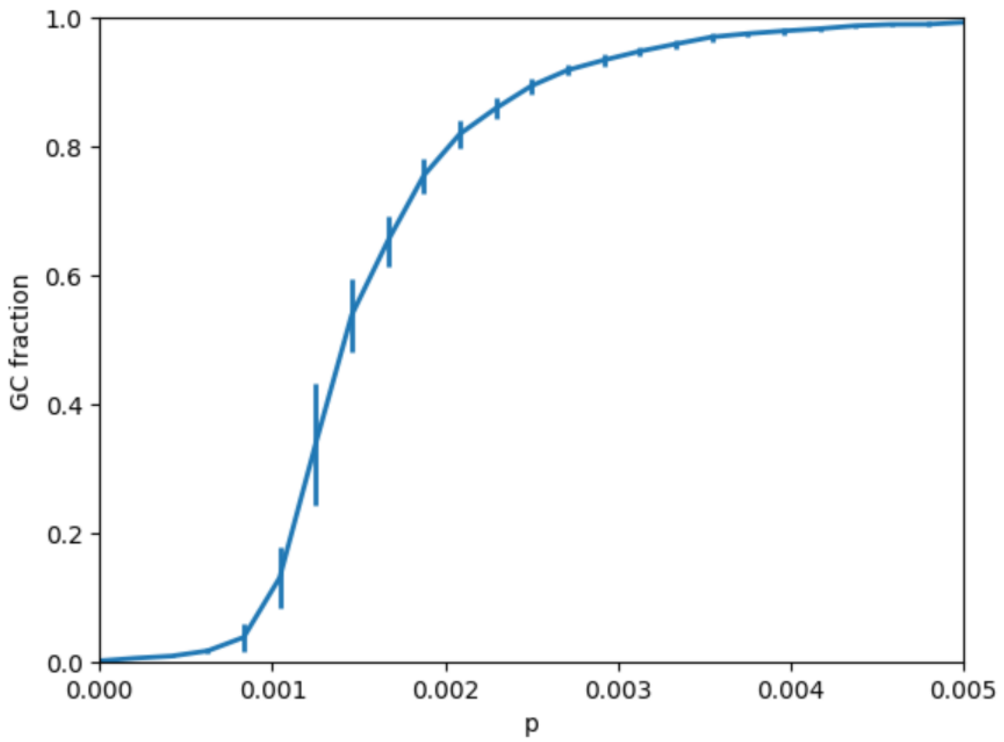
c. 250

d. 500

- $\langle l \rangle = \frac{\ln N - \gamma}{\ln \langle k \rangle} + \frac{1}{2}, \gamma \approx 0.577 \rightarrow N = 563$
- $\langle l \rangle \approx l_{max} \approx \ln N / \ln \langle k \rangle \rightarrow N \approx \langle k \rangle^{\langle l \rangle} = 1000$

CHAPTER 5 EXERCISES

5.10 Reproduce the plot of Figure 5.2 for networks with 1000 nodes. (*Hint*: Use the NetworkX function to generate random networks.) Use 25 equally spaced values of the link probability, in the interval $[0, 0.005]$. For each value generate 20 different networks, compute the relative size of the giant component and report the average and the standard deviation in the plot.

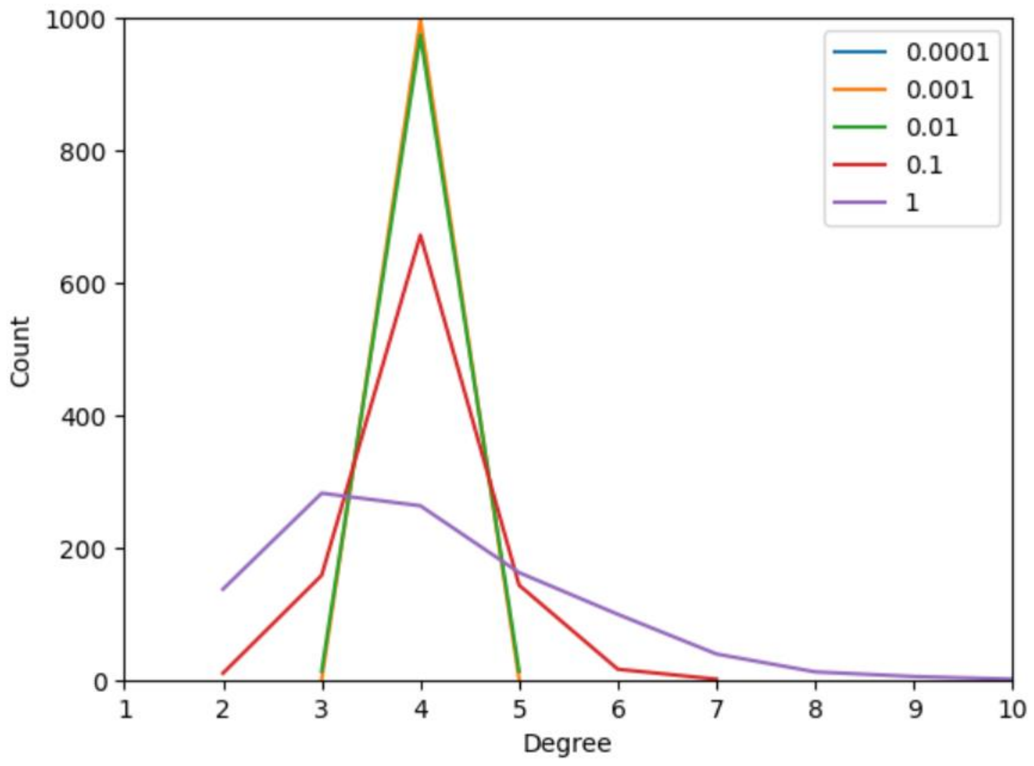



```python
# Plots the fraction of nodes in the giant component as a function of p

N = 1000
p = np.linspace(0, 0.005, 25)
reps = 20

results = dict.fromkeys(p)

for pi in tqdm(p):
    results_run = []
    for i in range(reps):
        G =  nx.gnp_random_graph(1000 , pi)
        GC_fraction = len(max(nx.connected_components(G), key=len))/N
        results_run.append(GC_fraction)
    results[pi] = results_run

df = pd.DataFrame(results)
plt.errorbar(df.keys(), df.mean(), yerr=df.std(), linewidth=2)

plt.xlabel('p')
plt.ylabel('GC fraction')
plt.xlim(0, 0.005)
plt.ylim(0, 1)
```

✓ 21.6s Python

CHAPTER 5 EXERCISES

5.15 Build Watts–Strogatz networks with 1000 nodes, $k = 4$, and these values for the rewiring probability: $p = 0.0001, 0.001, 0.01, 0.1, 1$. Compute and compare their degree distributions, by plotting them in the same diagram.




```python
# Plots degree distribution of Watts-Strogatz graphs

N = 1000
k = 4
p = [0.0001, 0.001, 0.01, 0.1, 1]

results = dict.fromkeys(p)
for pi in tqdm(p):
    G = nx.watts_strogatz_graph(N, k, pi)
    results[pi] = [G.degree(n) for n in G.nodes()]

plt.figure()
for pi in p:
    counts = pd.Series(results[pi]).value_counts().sort_index()
    plt.plot(counts.index, counts.values, label=pi)
plt.legend()
plt.xlabel('Degree')
plt.ylabel('Count')
plt.xlim(1, 10)
plt.ylim(0, 1000);
```

✓ 0.2s Python

CHAPTER 5 EXERCISES

5.21 In the fitness model, suppose that the fitness of a node coincides with its degree. Could you guess what kind of degree distribution the resulting network will have? (*Hint*: The discussion on non-linear preferential attachment in Section 5.5 might help.)

- Fitness model: $\Pi = \eta_j k_k / \sum_l \eta_l k_l$
- If $\eta_j = k_j$, it becomes a non-linear PA model with $\alpha = 2$
- Few nodes hoard all the connections