Redes complejas

Práctica 4:

MUSI - Máster Universitario en Sistemas Inteligentes

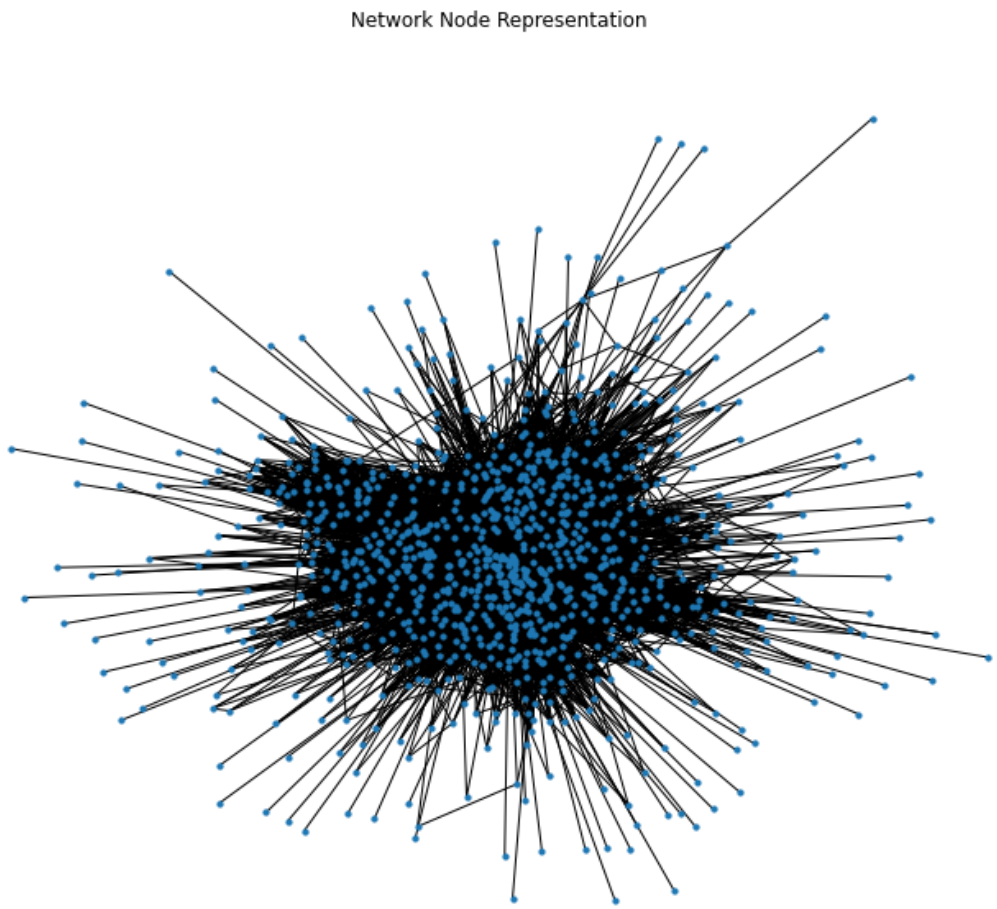
López Muñoz, Iván - 43221427B

Índice de contenidos

- **Seleccione una network real con mínimo 500 nodos y solo 1 componente**

La red se generó utilizando datos de correo electrónico de una gran institución de investigación europea (https://snap.stanford.edu/data/email-Eu-core-temporal.html). Se ha anonimizado la información sobre todos los correos electrónicos entrantes y salientes entre miembros de la institución de investigación. Los correos electrónicos solo representan la comunicación entre los miembros de la institución, y el conjunto de datos no contiene mensajes entrantes ni salientes hacia el resto del mundo. Una arista dirigida (u, v) significa que la persona u envió un correo electrónico a la persona v. Se crea una arista independiente para cada destinatario del correo electrónico. De esta forma se crea un único componente en toda la red:

Number of nodes: 986
Number of edges: 16064



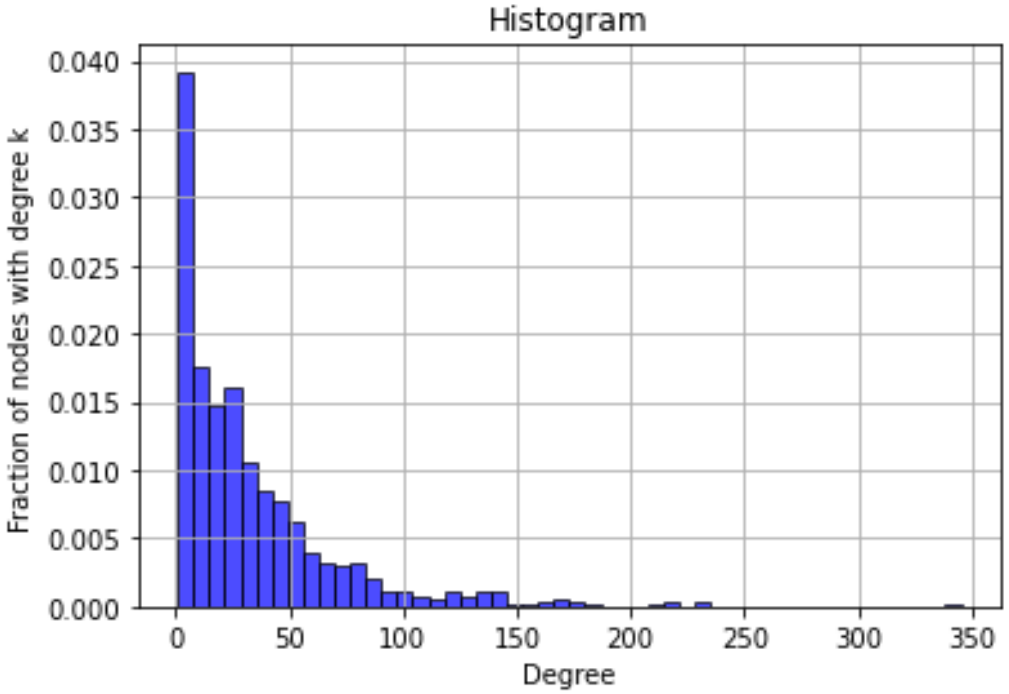
Código para generar la representación de la red

```
# Load the dataset
file_path = 'email-Eu-core-temporal.txt'
graph = nx.Graph()

# Read the dataset and add edges to the graph
with open(file_path, 'r') as file:
    for line in file:
        src, tgt, _ = map(int, line.strip().split())
        graph.add_edge(src, tgt)

# Plot 0 - Network Node Representation
plt.figure(figsize=(10, 8))
pos = nx.spring_layout(graph, seed=42)
nx.draw(graph, pos, with_labels=False, node_size=10)
```

```
plt.title("Network Node Representation")
plt.show()
```

1. **Imprima un histograma para la distribución de grados de los nodos**

Se obtiene, calculando la distribución de los nodos, el siguiente histograma:



Podemos observar en la cola del histograma como algunos nodos llegan a ser de grado 350 aprox. Para confirmarlo, calculamos la centralidad de los nodos en orden ascendente según el grado y obtenemos:

Node 90: Degree 345
Node 120: Degree 232
Node 772: Degree 231
Node 214: Degree 219
Node 951: Degree 216
Node 159: Degree 214
Node 607: Degree 183
Node 362: Degree 178
Node 890: Degree 175
Node 61: Degree 171

Por lo que, efectivamente el nodo 90 sería el último de la cola en el histograma, siendo el nodo con mayor número de conexiones seguido del resto.

Código para generar el histograma

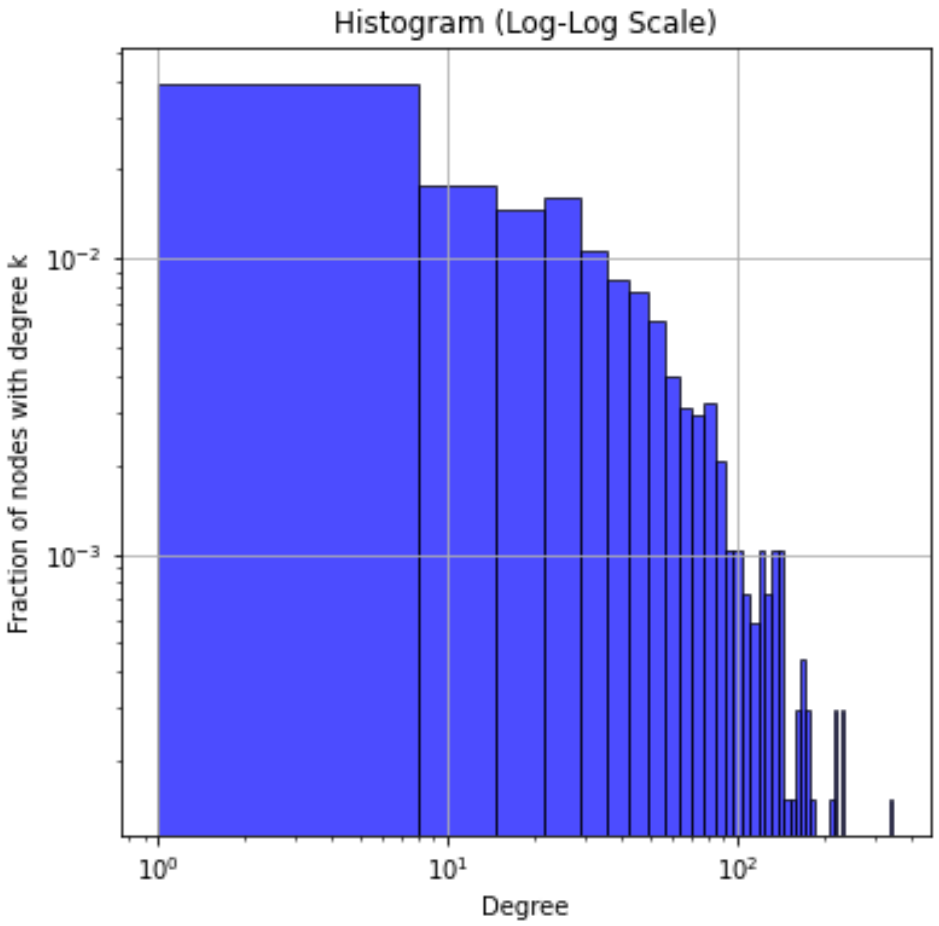
```
# Calculate the degree distribution
degree_sequence = [d for n, d in graph.degree()]

# Plot 1: Histogram
plt.hist(degree_sequence, bins=50, alpha=0.7, color='b', edgecolor='k', density=True)
plt.title('Histogram')
plt.xlabel('Degree')
plt.ylabel('Fraction of nodes with degree k')
plt.grid(True)
plt.show()
```

Código para obtener la información

```
# Get the degrees of all nodes in the graph
node_degrees = list(graph.degree())

# Sort nodes by their degrees in descending order
sorted_nodes = sorted(node_degrees, key=lambda x: x[1], reverse=True)

# Print the nodes with the highest degrees (top 10)
top_nodes = sorted_nodes[:10]  # You can change the number to print more or fewer nodes

for node, degree in top_nodes:
    print(f"Node {node}: Degree {degree}")
```

2. **Repita el trazado del histograma mediante escala log-log**

Se obtiene, cambiando la escala de los ejes a escala logarítmica, el siguiente histograma:



Un problema que nos encontramos en los dos histograma anteriores es que la información es insuficiente en la cola, es decir, en la región de valores altos de k, que es precisamente la región en la que la ley de potencia suele seguirse con mayor precisión.

Código para generar el histograma

```
# Plot 2: Histogram in log-log scale
plt.hist(degree_sequence, bins=50, alpha=0.7, color='b', edgecolor='k', density=True)
plt.title('Histogram (Log-Log Scale)')
plt.xlabel('Degree')
plt.ylabel('Fraction of nodes with degree k')
plt.xscale('log')
plt.yscale('log')
plt.grid(True)
plt.show()
```

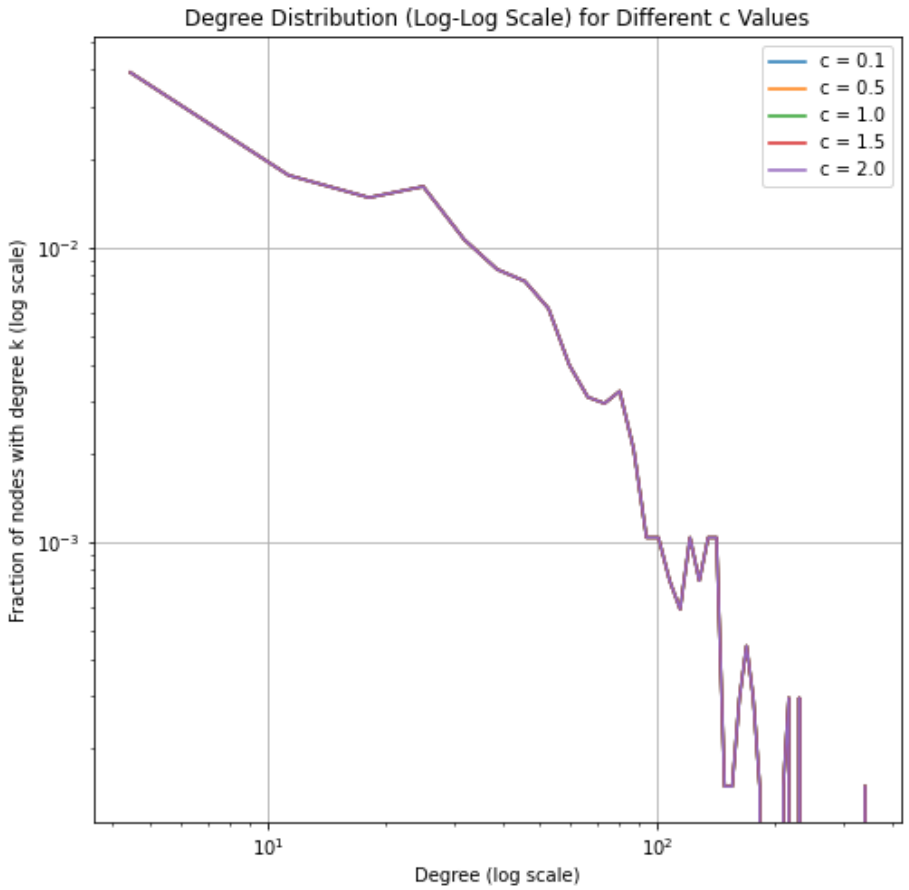
3. Representar la ecuación $p_k=Ck^{-\alpha}$ para diferentes valores de C y α

En este apartado se pretende observar cómo varía la ley de potencia según el valor de las constantes C y α.

En este caso: $C = e^{c}$ y $\alpha = 1 + N(\Sigma\, ln\, \frac{ki}{kmin - 0.5})^{-1}$

siendo α la exponente de la ley de potencia, C una constante que se arregla con el requisito de la normalización, N el número de nodos con grado mayor o igual que k_{min} y k_{min} es el grado mínimo para el cual se cumple la ley de potencia.

En este caso podemos conocer el valor de α, por lo que se van a generar diferentes gráficos según la constante C:



Se observa como los valores de c no tienen efecto en el cálculo de la distribución según la ley de potencia.

Código para generar el gráfico

```
# Plot 3: Create a plot with the degree distribution in log-log scale
plt.figure(figsize=(8, 8))

# Calculate alpha and kmin
kmin = min(degree_sequence)
N = sum(1 for k in degree_sequence if k >= kmin)
alpha = 1 + N * sum(np.log(k / (kmin - 0.5))**-1 for k in degree_sequence if k >= kmin)

# Define a range of 'c' values
c_values = [0.1, 0.5, 1.0, 1.5, 2.0]

for c in c_values:
    C = e**c
    pk = [C * (k**(-alpha)) for k in degree_sequence]

    # Plot the degree distribution using the formula in log-log scale
    hist, bin_edges = np.histogram(degree_sequence, bins=50, density=True)
    bin_centers = 0.5 * (bin_edges[:-1] + bin_edges[1:])
    plt.loglog(bin_centers, hist, label=f'c = {c}')

plt.title('Degree Distribution (Log-Log Scale) for Different c Values')
plt.xlabel('Degree (log scale)')
plt.ylabel('Fraction of nodes with degree k (log scale)')
plt.legend()
plt.grid(True)
plt.show()
```

4. **Aplique un agrupamiento logarítmico para una mejor visualización del histograma**

Para solucionar el problema de la “señal ruidosa” en la cola de la distribución, podemos aplicar un agrupamiento logarítmico. Este agrupamiento nos va a permitir poder mejorar la visualización o detectar el comportamiento de la ley de potencia.

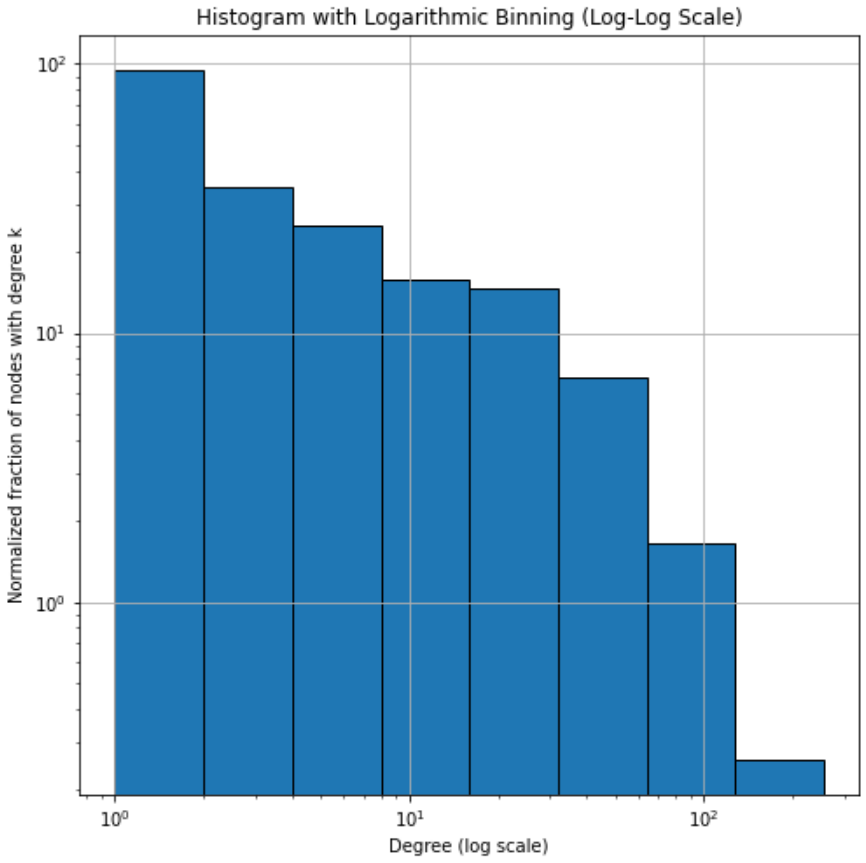
La forma más sencilla para aplicar un agrupamiento logarítmico es utilizar un histograma con barras más grandes, de modo que haya más muestras en cada barra, produciendo menos ruido, pero a expensas de menos detalles en general, ya que el número de barras se reduce considerablemente.

La anchura de las barras se va a multiplicar por una constante ‘a’ para que cada una cubra un intervalo de la distribución del eje de los grados.

Los intervalos de los grados ‘k’ se rigen por lo siguiente:

$a^{n-1} \leq k < a^{n}$ con una anchura de las barras de $a^{n} - a^{n-1}$

La opción más común es: a = 2



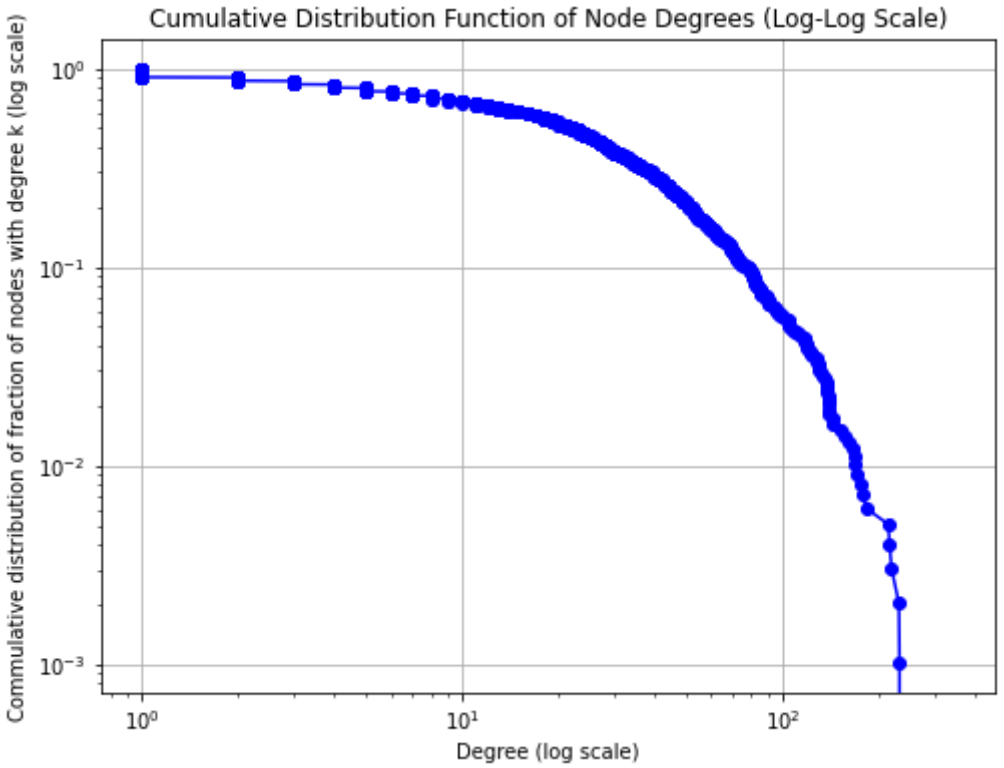
Código para generar el histograma

```
# Plot 4: Create a single plot with the degree distribution in log-log scale
plt.figure(figsize=(8, 8))

# Define the number of bins and the range
num_bins = 50
min_degree = min(degree_sequence)
max_degree = max(degree_sequence)

# Choose the base 'a' for logarithmic binning (e.g., a=2)
a = 2

# Calculate bin edges based on logarithmic binning
bin_edges = [a ** n for n in range(int(np.log(min_degree) / np.log(a)), int(np.log(max_degree)
/ np.log(a)) + 1)]
widths = [bin_edges[i + 1] - bin_edges[i] for i in range(len(bin_edges) - 1)]

# Calculate the histogram with logarithmic binning
hist = np.histogram(degree_sequence, bins=bin_edges)
# Normalize by bin width
hist_norm = hist[0] / widths
# Plot the histogram with logarithmic binning and black borders
plt.bar(bin_edges[:-1], hist_norm, widths, align='edge', edgecolor='black')
plt.xscale('log')
plt.yscale('log')
plt.title('Histogram with Logarithmic Binning (Log-Log Scale)')
plt.xlabel('Degree (log scale)')
plt.ylabel('Normalized fraction of nodes with degree k')
plt.grid(True)
plt.show()
```

5. **Representa la función de distribución acumulativa de los grados de nodos de la red.**

Otra forma de visualizar la distribución de la ley de potencia es calculando la distribución acumulada con escalas logarítmicas.

La distribución acumulativa, a comparación con los histogramas, preserva toda la información contenida en los datos, porque no hay barras involucradas. La manifestación más obvia de esta diferencia es el número de puntos representados en la gráfica. Además que se observa con mayor claridad la ley de potencia en la distribución:



Código para generar el gráfico

```
# Plot 5: cummulative distribution function for the degrees of nodes on the network
# Sort the degrees in ascending order
sorted_degrees = np.sort(degree_sequence)

# Calculate the CDF values
cdf = np.arange(1, len(sorted_degrees) + 1) / len(sorted_degrees)
```

```
# Plot the CDF on logarithmic scales
plt.figure(figsize=(8, 6))
plt.loglog(sorted_degrees, 1 - cdf, marker='o', linestyle='-', color='b')
plt.grid(True)

plt.title('Cumulative Distribution Function of Node Degrees (Log-Log Scale)')
plt.xlabel('Degree (log scale)')
plt.ylabel('Commulative distribution of fraction of nodes with degree k (log scale)')
plt.show()
```

6. **Concluye con si la red seleccionada es una red libre de escala o no**

La determinación de si una red es una red libre de escala depende de las características de su distribución de grados. En una red libre de escala, la distribución de grados sigue una distribución de ley de potencia, lo que implica que existe un reducido número de nodos con grados muy elevados, mientras que la mayoría de los nodos poseen grados relativamente bajos.

Para concluir si la red seleccionada es o no una red libre de escala, es necesario analizar su distribución de grados. Si la distribución de grados presenta un comportamiento de ley de potencia, manifestándose como una línea recta en una gráfica log-log, esto sugiere que la red sigue una topología libre de escala. No obstante, si la distribución de grados no sigue una distribución de ley de potencia, ello indica que la red carece de características propias de una red libre de escala.

Es importante tener en cuenta que en el mundo real, las redes rara vez exhiben una distribución de ley de potencia perfecta debido a diversos factores y limitaciones. La determinación de si una red es libre de escala es, en cierta medida, subjetiva y depende de las particularidades de la red y la calidad de los datos.

En el caso de este ejercicio, al observar el gráfico del ejercicio 3, podemos concluir que la red presenta este fenómeno.