UAS - Assignment 1 (Module 2)

Sreeja Guduri (2021102007)

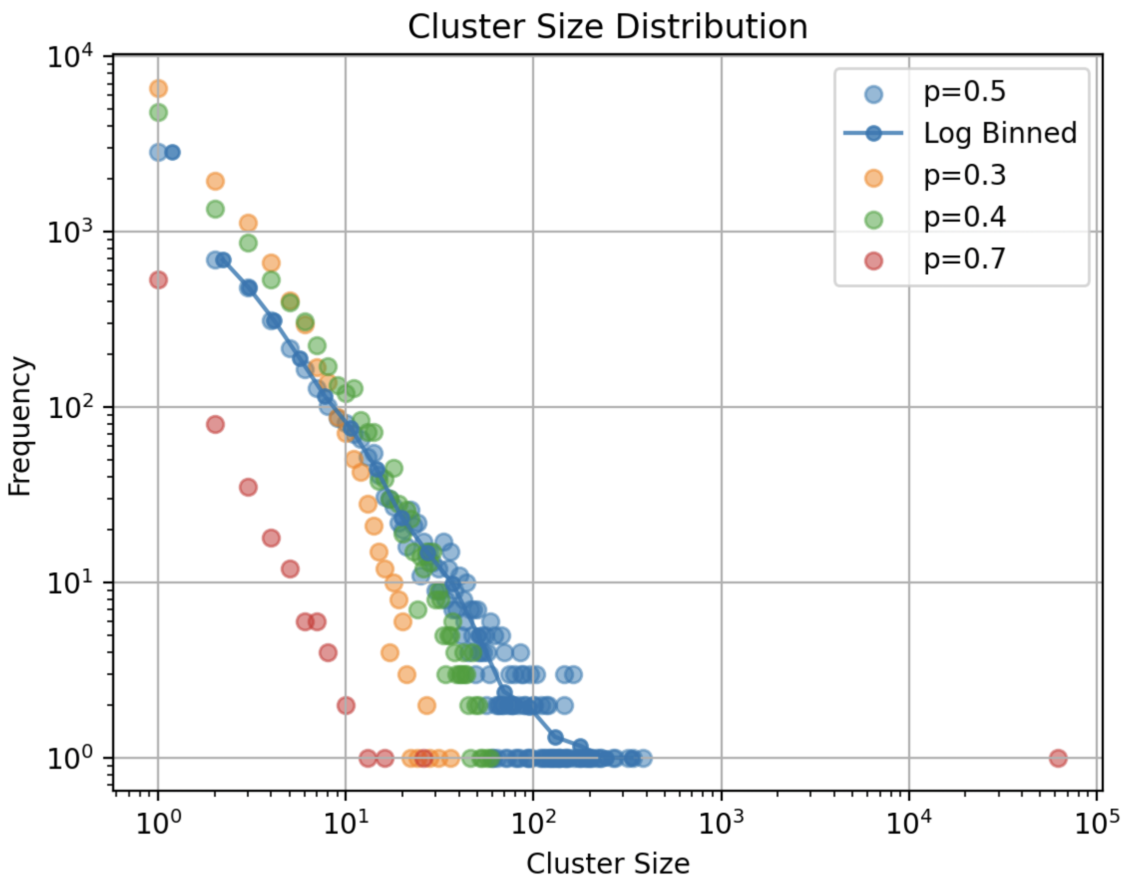
Question - 1

- p = 0.3 represents the subcritical region. Here the clusters will be small and scattered.
- p = 0.4 is when we are approaching criticality (pc = 0.5) where larger clusters start forming
- p = 0.5 is the critical point when the system undergoes a phase transition. The cluster size now follows the power law and the largest cluster spans the whole lattice.
- p = 0.7 represents the subcritical regime where there is a high probability that a large, spanning cluster exists.

The below code is used to generate the graphs that represent the 2D lattice:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import stats  # Add this line for scipy.stats

def percolation(n, p):
    # Create an empty lattice
    lattice = np.zeros((n, n))

    for i in range(n):
        for j in range(n):
            if np.random.random() <= p:
                lattice[i, j] = 1

    def dfs(i, j):
        stack = [(i, j)]
        cluster = []
        while stack:
            x, y = stack.pop()
            if lattice[x, y] == 1:
                cluster.append((x, y))
                lattice[x, y] = 0  # Mark as visited
                for dx, dy in [(1, 0), (-1, 0), (0, 1), (0, -
                    nx, ny = x + dx, y + dy
                    if 0 <= nx < n and 0 <= ny < n:
                        stack.append((nx, ny))
        return cluster

    clusters = []
    for i in range(n):
        for j in range(n):
            if lattice[i, j] == 1:
                cluster = dfs(i, j)
                clusters.append(len(cluster))

    return clusters

# Parameters
Lsize = 300
pc = 0.5

# Probabilities to simulate
probabilities = [pc, 0.3, 0.4, 0.7]

# Plot
plt.figure()
for p in probabilities:
    clusters = percolation(Lsize, p)
    unique_sizes, counts = np.unique(clusters, return_counts=
    plt.scatter(unique_sizes, counts, label=f'p={p}', alpha=0

    if p == pc:#unique_sizes, counts, bin_count=20
        bin_count = 20
        bins = np.logspace(np.log10(min(unique_sizes)), np.lo
        bin_means, bin_edges, _ = stats.binned_statistic(uniq
        bin_centers = (bin_edges[1:] + bin_edges[:-1]) / 2
        plt.plot(bin_centers, bin_means, linestyle='-', marke

plt.xscale('log')
plt.yscale('log')
plt.xlabel('Cluster Size')
plt.ylabel('Frequency')
plt.title('Cluster Size Distribution')
plt.legend()
plt.grid(True)
plt.show()
```

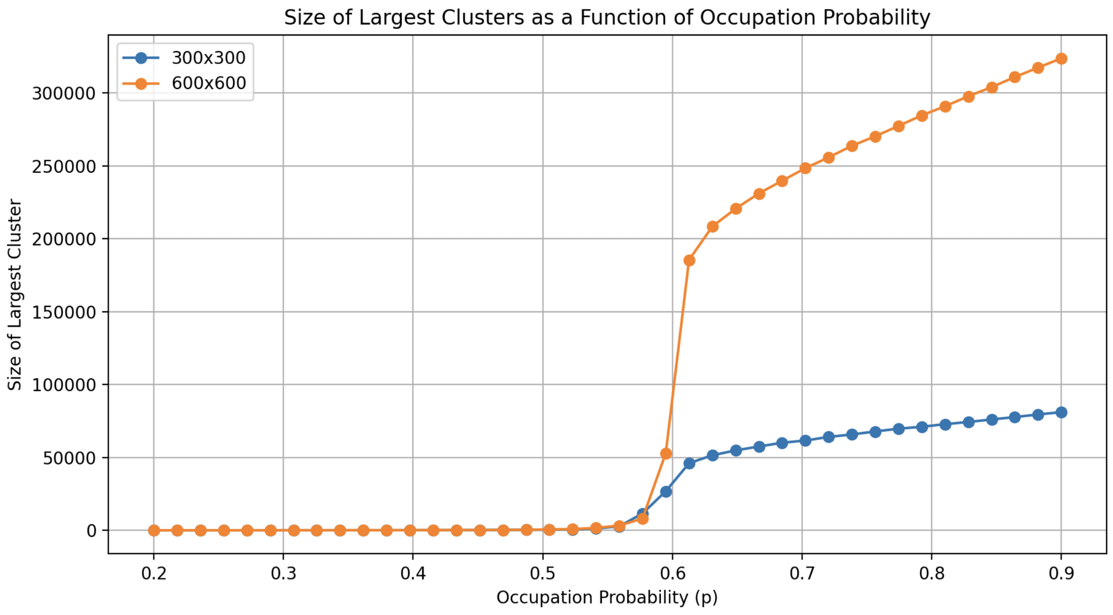
The output graph we get is:



Question - 2

The graphs generated in this question indicate that as the cluster size increases, the probability of forming a larger cluster also increases.

```
import numpy as np
import matplotlib.pyplot as plt

def generate_matrix(p_values):
    for p in p_values:
        lattice = np.random.rand(L, L) < p
        labels = np.zeros_like(lattice, dtype=int)

        cluster_label = 1

        for i in range(L):
            for j in range(L):
                if lattice[i, j]:
                    #  label clusters
                    top = labels[i-1, j] if i > 0 else 0
                    left = labels[i, j-1] if j > 0 else 0

                    if top == 0 and left == 0:
                        labels[i, j] = cluster_label
                        cluster_label += 1

                    elif top == 0:
                        labels[i, j] = left

                    elif left == 0:
                        labels[i, j] = top

                    elif top != left:
                        min_label = min(top, left)
                        max_label = max(top, left)
                        labels[labels == max_label] = min_labe
                        labels[i, j] = min_label
                    else:
                        labels[i, j] = top

        unique_labels, label_counts = np.unique(labels, retur
        label_counts = label_counts[1:]
        largest_cluster_size = label_counts.max()
        largest_cluster_sizes.append(largest_cluster_size)

    return largest_cluster_sizes



p_critical = 0.5
p_values = np.linspace(0.2, 0.9, 40)
L_values = [300, 600]

plt.figure()

for L in L_values:
    largest_cluster_sizes = []
    largest_cluster_sizes = generate_matrix(p_values)
    plt.plot(p_values, largest_cluster_sizes, label=f'{L}x{L}

plt.xlabel('Occupation Probability (p)')
plt.ylabel('Size of Largest Cluster')
plt.title('Size of Largest Clusters as a Function of Occupati
plt.legend()
plt.grid(True)
plt.show()
```



Question - 3

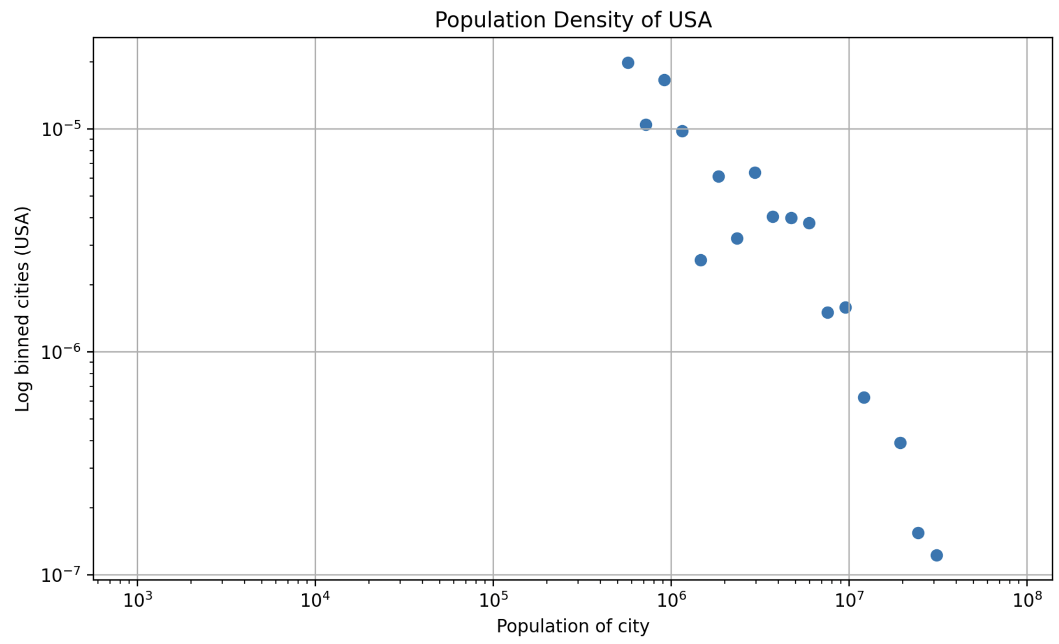
The datasets used were of Indian and American cities and a histogram representing that data has been plotted. From the histogram it is apparent that both the distributions follow the power law (exponent relationship) between the population of the city and the cities in the country.

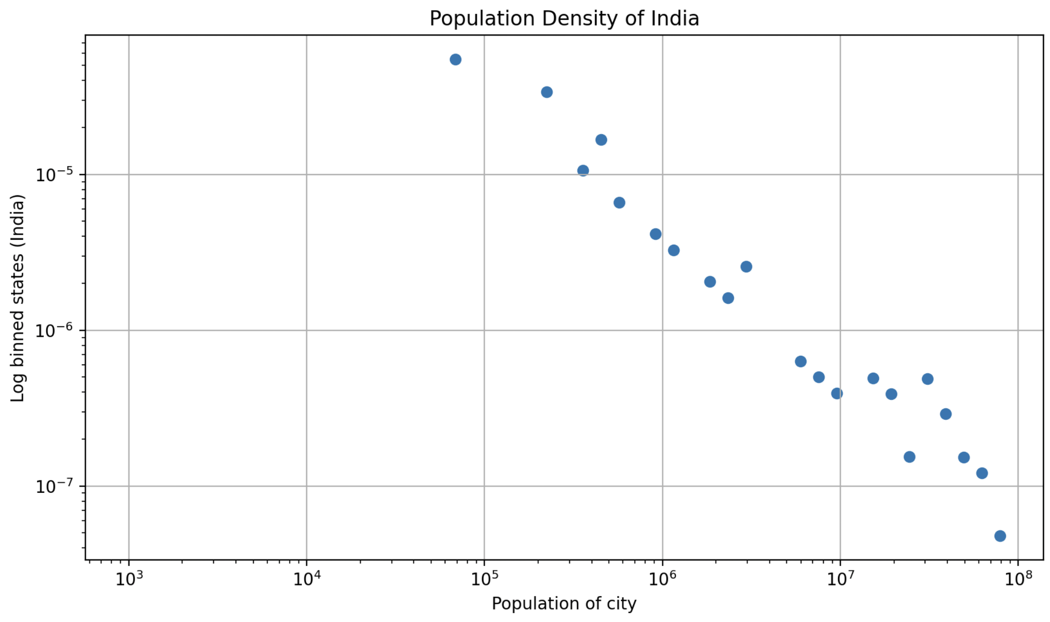
The code used is:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats

usa_df = pd.read_csv("usa.csv")
india_df= pd.read_csv("india.csv")
print(usa_df)

usa_df['Population'] = usa_df['Population'].str.replace(',',
bins = np.logspace(3, 8, 50)
widths = (bins[1:] - bins[:-1])
```

```
hist = np.histogram(usa_df['Population'], bins=bins)
hist_norm = hist[0]/widths
hist_per = hist[0]/sum(hist[0])
hist_norm_per = hist[0] / (widths * sum(hist[0]))
plt.figure()
plt.scatter(bins[:-1], hist_norm)
plt.xscale('log')
plt.yscale('log')
plt.xlabel("Population of city")
plt.ylabel("Log binned cities (USA)")
plt.title('Population Density of USA')
plt.grid(True)
plt.show()
plt.show()



print(india_df)
india_df['Population'] = india_df['Population'].str.replace('
bins = np.logspace(3, 8, 50)
widths = (bins[1:] - bins[:-1])
hist = np.histogram(india_df['Population'], bins=bins)
hist_norm = hist[0]/widths
hist_per = hist[0]/sum(hist[0])
hist_norm_per = hist[0] / (widths * sum(hist[0]))
plt.figure()
plt.scatter(bins[:-1], hist_norm)
plt.xscale('log')
plt.yscale('log')
plt.xlabel("Population of city")
plt.ylabel("Log binned states (India)")
plt.title('Population Density of India')
plt.grid(True)
plt.show()
plt.show()
```

The histograms generated are:


Population Density of USA
Log binned cities (USA)
Population of city
10^{-5}
10^{-6}
10^{-7}
10^3
10^4
10^5
10^6
10^7
10^8


Population Density of India
Log binned states (India)
Population of city
10^{-5}
10^{-6}
10^{-7}
10^3
10^4
10^5
10^6
10^7
10^8