Task 4

April 2024

Understand common classical network models: small world network, rule network, random network, familiar with the principles of common classical network model algorithms and realize them through Python language programming simulation.

The small-world network model refers to a network with a short average path length and a high clustering coefficient.

The small-world network model is mainly determined by the average path length L and the clustering coefficient C.

The average path length L (also known as the characteristic road length) is defined as the average value of the distance between any two nodes (excluding the self node to the self node) in the network, and its expression formula is as follows:

$$L = \frac{\sum_{i \neq j} d_{ij}}{1/2 \times N(N-1)}$$

N is the number of network nodes, dij, which is the number of edges on the shortest path between link node i and node j.

The network clustering coefficient Ci is defined as the average of the clustering coefficients of each node, and the clustering coefficient of a node i is defined as Ci the average of the number of edges ki between all adjacent points linked to node Ei, and their expressions are as follows:

$$C = \frac{\sum_{i=1}^{N} C_i}{N}$$

$$C_i = \frac{E_i}{1/2 \times k_i \times (k_i - 1)}$$

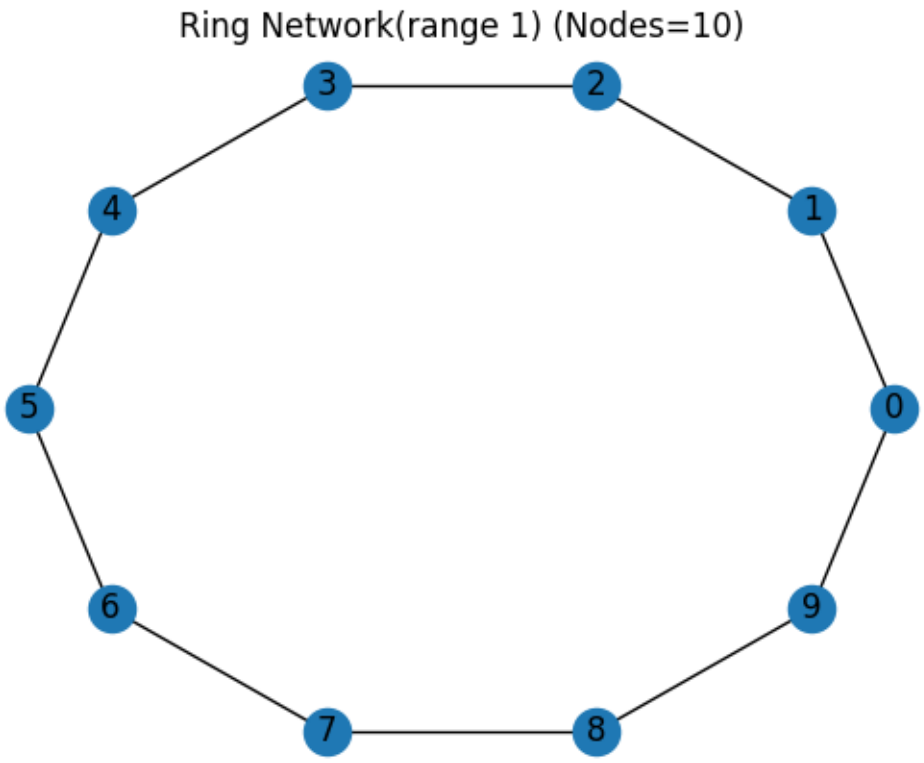
A regular network is a type of network in which the relationships between nodes can be represented by some regular structure. There is an established rule relationship between any two nodes in the network. A regular network is generated when the reconnection probability p is 0 .

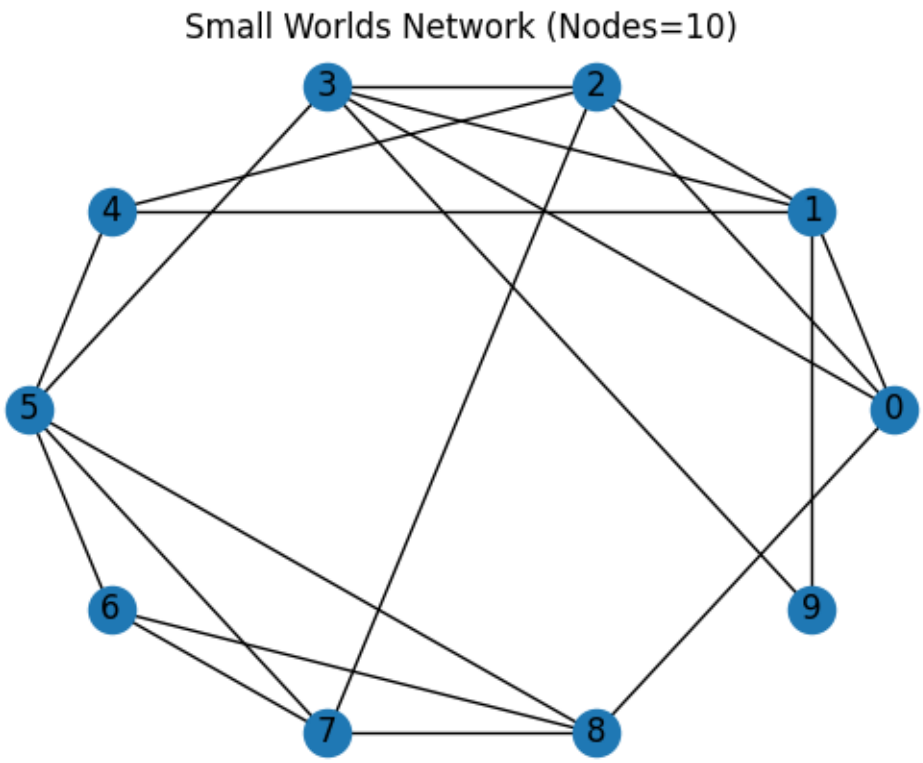
The characteristics of regular network can be observed through the degree distribution of each node.

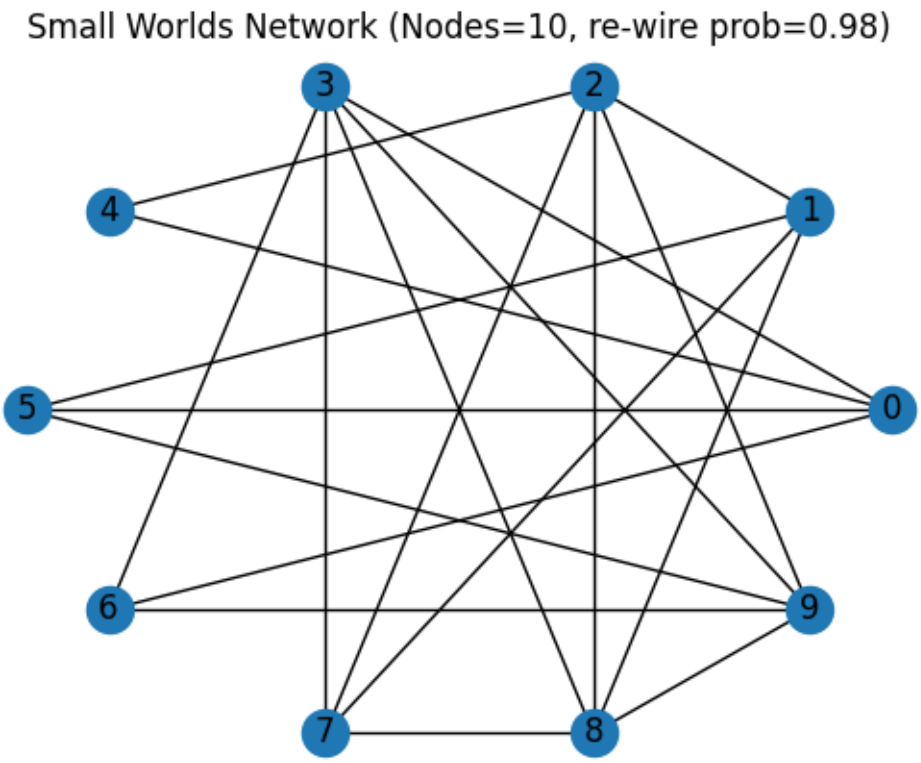
Random network is a kind of network where nodes are not linked according to established rules, but randomly linked to each other with a certain probability.

Random network and regular network are two opposite cases, and the generated network is a random network when the reconnection probability p is 1 .

The degree distribution of stochastic network model has Poisson distribution tendency.
The following uses python programming simulation:




Small Worlds Network (Nodes=10)
3
2
4
1
5
0
6
9
7
8


Small Worlds Network (Nodes=10, re-wire prob=0.98)
3
2
4
1
5
0
6
9
7
8

```
import networkx as nx
import matplotlib.pyplot as plt
import random
import sys

def create_random_network(N, p):
    G = nx.Graph()
    for i in range(N):
        for j in range(i + 1, N):
            if random.random() < p:
                G.add_edge(i, j)
    return G

def create_ring_network(N):
    G = nx.cycle_graph(N)
    return G

def create_small_world_network(N, p=0.2):
    G = nx.watts_strogatz_graph(N, k=4, p=p)
    return G

def draw_circular_graph(G, title):
    pos = nx.circular_layout(G)
    plt.title(title)
    nx.draw(G, pos, with_labels=True)
    plt.show()
if __name__ == "__main__":
    if len(sys.argv) == 3 and sys.argv[1] == "-ring_network":
        N = int(sys.argv[2])
        ring_network = create_ring_network(N)
        draw_circular_graph(ring_network,
        f"Ring Network(range 1)
--------(Nodes={N})")
    elif len(sys.argv) == 3 and sys.argv[1] == "-small_world":
        N = 10  #
        small_world_network = create_small_world_network(N)
        draw_circular_graph(small_world_network,
        f"Small Worlds Network (Nodes={N})")
    elif len(sys.argv) == 5 and sys.argv[1] == "-small_world"
    and sys.argv[3] == "-re_wire":
        N = int(sys.argv[2])
        p = float(sys.argv[4])
        small_world_network = create_small_world_network(N, p)
        draw_circular_graph(small_world_network,
        f"Small Worlds Network (Nodes={N}, re-wire prob={p})")
    elif len(sys.argv) == 4 and sys.argv[1] == "-random_network":
```

```
            N = int(sys.argv[2])
            p = float(sys.argv[3])
            random_network = create_random_network(N, p)
            draw_circular_graph(random_network,
            f"Random Network
--------(Nodes={N}, re-wire prob={p})")
    else:
            print("Usage:")
            print("For ring network:
--------python3 assignment.py --ring_network <N>")
            print("For default small world network:
--------python3 assignment.py --small_world <N>")
            print("For custom small world network
--------with re-wiring probability:
--------python3 assignment.py --small_world <N> --re_wire <probability>")
            print("For random network:
--------python3 assignment.py --random_network <N> <p>")
#TEST
# python assignment.py --ring_network 10
# python assignment.py --small_world 10
# python assignment.py --small_world 10 --re_wire 0.98
# python assignment.py --random_network 10 0.5
```