# 1.Basic

## 1.1 Read and write graphs

import sys

import matplotlib.pyplot as plt

import networkx as nx

g = nx.grid\_2d\_graph(5, 5) # 5\*5 grid

# print the adjacency(n. 毗邻；四周；邻接物) list

for line in nx.generate\_adjlist(g):

print(line)

#write edgelist to grid.edgelist

nx.write\_edgelist(g, path="grid.edgelist", delimiter=":") # delimiter:定界符，在grid.list里面可以看到":

#read edgelist from grid.edgelist

H = nx.read\_edgelist(path="grid.edgelist", delimiter=":")

nx.draw(H)

plt.show()

## 1.2 Peoperties

import matplotlib.pyplot as plt

from networkx import nx

g = nx.lollipop\_graph(4, 6) #棒棒糖图，lollipop graph ， 4个节点的圈和6个节点的棒（图中可见）

pathlengths = [] #创建一个空列表

print("source vertex {target:length,}") #target：目标；靶子

for v in g.nodes():

spl = dict(nx.single\_source\_shortest\_path\_length(g, v))

print('{} {}'.format(v, spl)) #{}：占用一个空格，然后按照后面的格式分别将v,spl填进去

for p in spl:

pathlengths.append(spl[p])

print('')

#取出每条路径，计算平均值

print("average shortest path length %s" %(sum(pathlengths)/len(pathlengths))) #%s：通过str() 字符串转换来格式化

#histogram(直方图) of path lengths ,

dist = {} #创建一个空字典，

for p in pathlengths:

if p in dist:

dist[p] += 1 #路径存在在原来基础上加1

else:

dist[p] = 1 #路径不存在设为1，

print('') # 打印空格

print("length #paths")

verts = dist.keys()

#sort 是应用在 list 上的方法，sorted 可以对所有可迭代的对象进行排序操作。list 的 sort 方法返回的是对已经存在的列表进行操作，

#而内建函数 sorted 方法，返回的是一个新的 list，而不是在原来的基础上进行的操作。

for d in sorted(verts):

print('%s %d ' % (d, dist[d])) #d%：有符号，十进制

print("radius: %d" % nx.radius(g)) #半径

print("diameter: %d "% nx.diameter(g)) #直径

print("eccentricity: %s" % nx.eccentricity(g)) # n. [数] 离心率；古怪；怪癖；

print("center: %s " % nx.center(g)) #中心

print("periphery: %s " % nx.periphery(g)) # 边缘，外围

print("density: %s " % nx.density(g)) #密度

nx.draw(g, with\_labels=True)

plt.show()