建立传播模型 与真实世界对比 能够准确描述真实世界例子

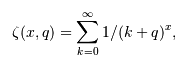
已知真实世界 每秒感染的节点数量，感染数量是随着时间增加越来越多的，但是，增加是有规律的， 有接触才能传染，

假如已知一个接触关系图，讨论在有感染源的情况下，节点状态随时间变化的情况，

根据Cckun模型，得出了感染百分比为0.46%，不随时间t而变化，将节点转换概率与感染概率分开，节点有易感染状态到感染状态，与周围邻居有关，而节点是否感染，即节点的感染百分比，是统计规律，由实验数据所得，为0.46%。

from scipy.special import \*

f(x,2.28)=x\*\*(-2.28)/zeta(2.28,1)



在随机图模型(也称为ER随机图)里，网络中任两个节点之间的连接概率都是p。因此，如果一个ER随机图其包含的节点数为N，则此随机图包含的边数约为p{N(N一1)/2}。

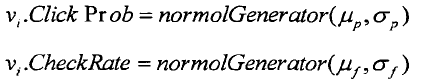
用户节点i查看邮箱时间设为T，打开可疑邮件概率设为Pi,i =1,2,...N，由于网络用户众多且各自操作行为相互独立，所以该模型假设不同用户查看邮箱的频率F与打开可疑邮件概率P均服从状态分布(均为正值)。

而对于某一用户i来说，在不同时刻查看邮箱的时间分布服从以Fi为均值的指数分布，即



用户i打开可疑邮件概率和查看邮箱的频率分别表示为:



下一阶段：看论文email14,sir9，SII20,sm21

产生随机数

Python自带的random库

1.参生n--m范围内的一个随机数: random.randint(n,m)

2.产生0到1之间的浮点数: random.random()

3.产生n---m之间的浮点数: random.uniform(1.1,5.4)

4.产生从n---m间隔为k的整数: random.randrange(n,m,k)

5.从序列中随机选取一个元素: random.choice([1, 2, 3, 4, 5, 6, 7, 8, 9, 0])

6.在一些特殊的情况下可能对序列进行一次打乱操作: random.shuffle([1,3,5,6,7])

7.random.sample(seq, k)实现从序列或集合seq中随机选取k个独立的的元素

for i in range(18):

    for nbr, datadict in G.adj.items():#nbr节点名称，datadict与节点相连边

        for i in I:

            i=str(i)

            if i ==nbr:

                for key in datadict:

                    key = int(key)

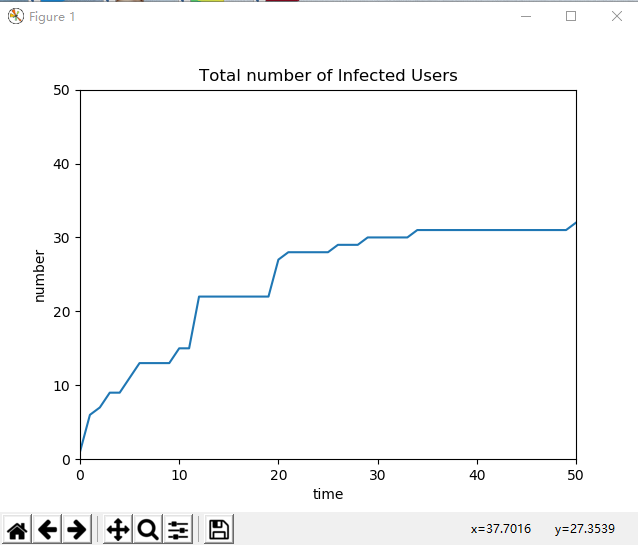
                    if key in S:

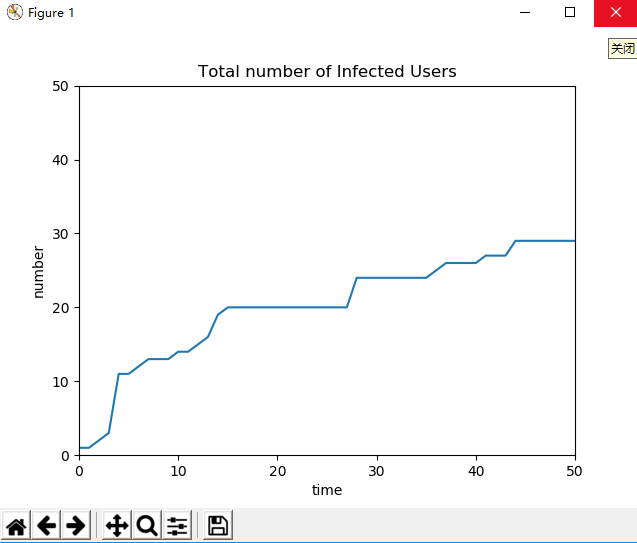
                        if random.random() <= 0.5:

                            I.append(key) #1.到第51行，此时I改变，不再是本轮循环的I

                            S.remove(key)

                count.append(len(I))





BA无标度网络与scale\_free网络不同，networkx中都可以直接生成。

用networkx可以直接生成scale\_free网络，但是是有向图，边表示为（0,1,0.5）第三个表示权重。

G = nx.scale\_free\_graph(400)

可以直接将有向图转化为 无向图

[Graph.to\_undirected()](https://www.osgeo.cn/networkx/reference/classes/generated/networkx.Graph.to_undirected.html#networkx.Graph.to_undirected) 或

H = nx.Graph(G)

400000个节点太大，直接用networkx画跑不出来，

400个节点时可以生成gexf格式，

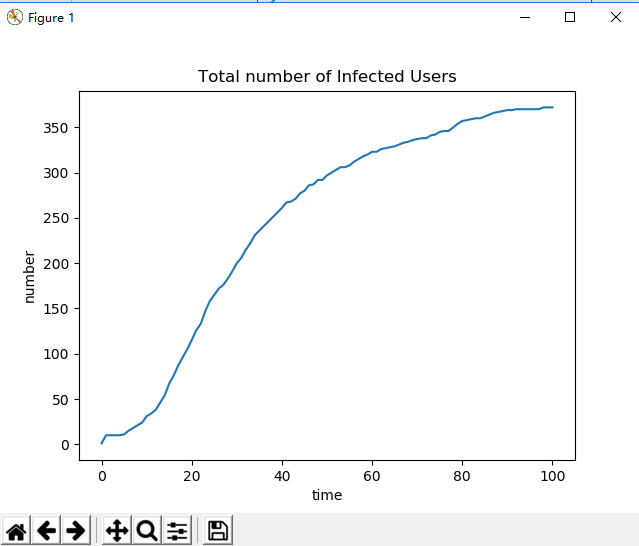
nx.write\_gexf(H,'un-scale-free.gexf')

在Gephi中显示出图的分布情况，并对该图进行处理：

<https://blog.csdn.net/u014595019/article/details/50759277>

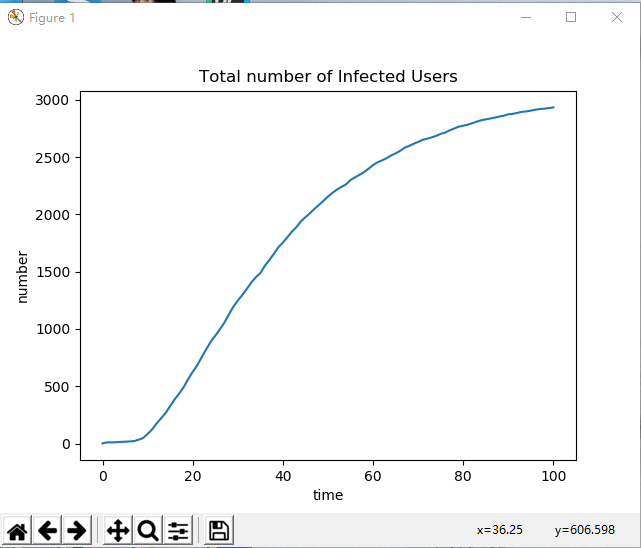
SMS4.py是该种情况下，不考虑免疫状态的程序。

（1）节点数4000，初始节点10个，感染概率0.046，共循环100轮，最终感染节点数372，感染过程如图：



（2）节点数40000，初始节点10个，感染概率0.046，共循环100轮，最终感染节点数2933，感染过程如图：

短时间内跑不出来



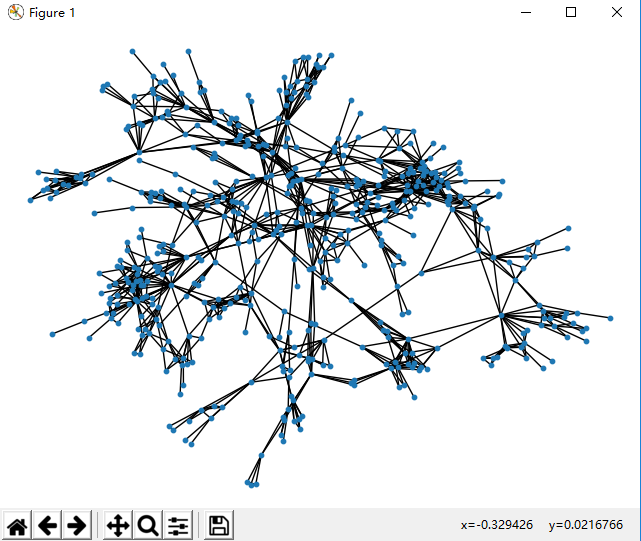
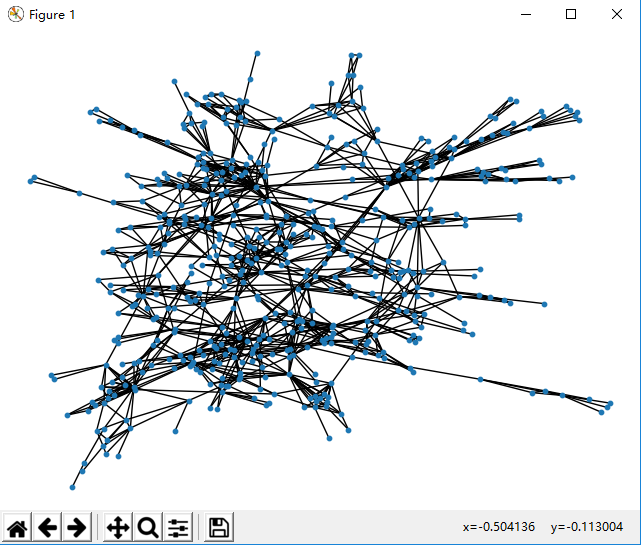
（3）考虑免疫状态试一下

28social\_network说明：

node = list(map(int,G.nodes))#图中节点列表，元素整数型

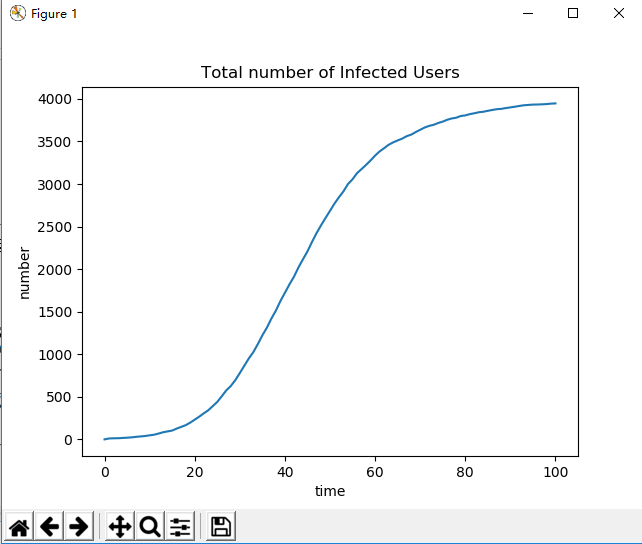
start\_node = random.choice(node)

随机选取node列表中的一个元素，直接用G.nodes不行。

右图为Spring布局

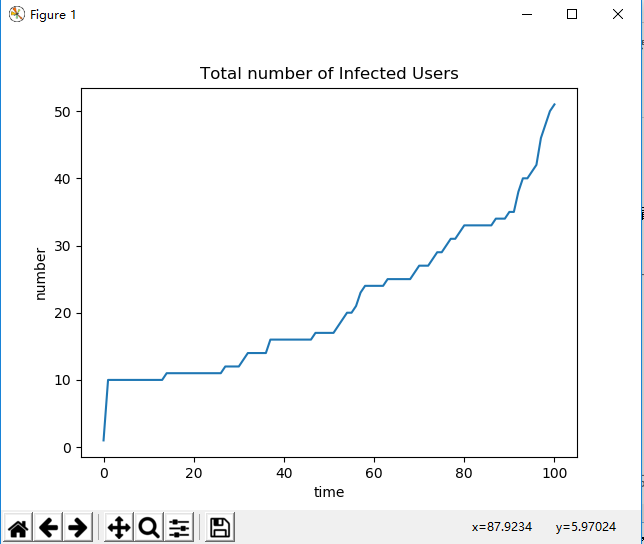
28social\_network.py

(1)4000个节点的传播网络，10个初始感染节点，感染概率为0.046（偏大），不考虑免疫状态，感染曲线如图：感染节点数为3947，时间（循环次数）为100，



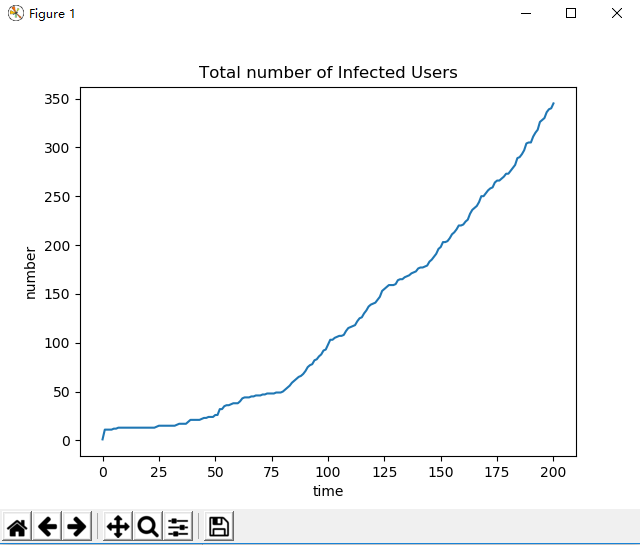
(2)4000个节点的传播网络，10个初始感染节点，感染概率为0.0046，考虑免疫状态：免疫概率0.0046，感染曲线如图：感染节点数为51，时间（循环次数）为100，

M:I:S=60:51:3949



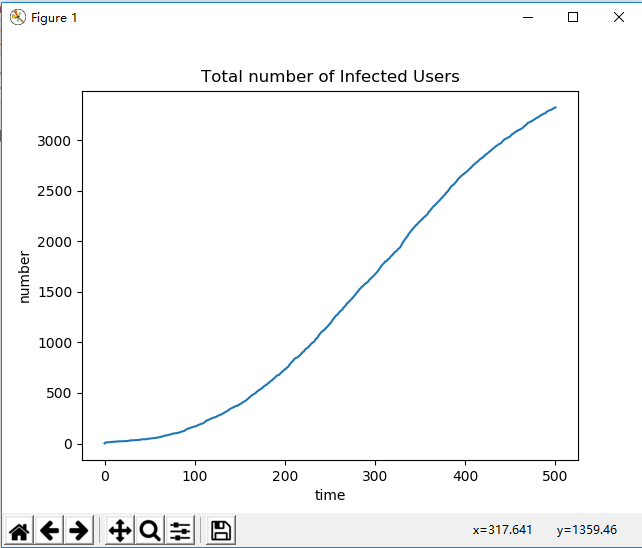
（3）4000个节点的传播网络，10个初始感染节点，感染概率为0.0046，考虑免疫状态：免疫概率0.0046，感染曲线如图：时间（循环次数）为200，

M:I:S=341:345:3655



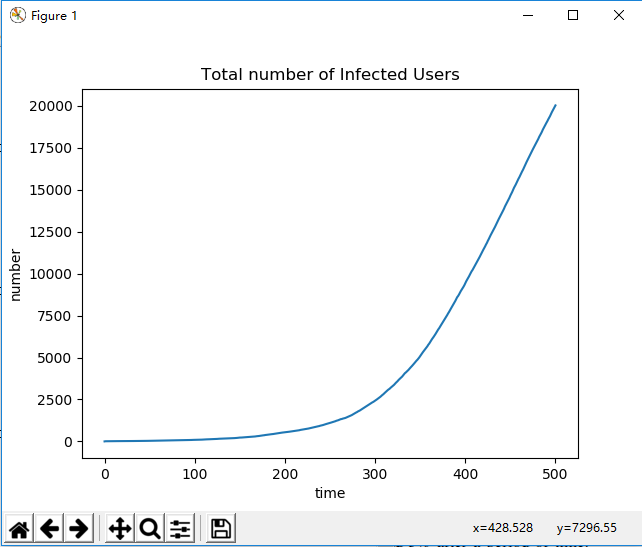
(4) 4000个节点的传播网络，10个初始感染节点，感染概率为0.0046，考虑免疫状态：免疫概率0.0046，时间（循环次数）为500，感染曲线如图：

节点数为M:I:S=3295：3323：677



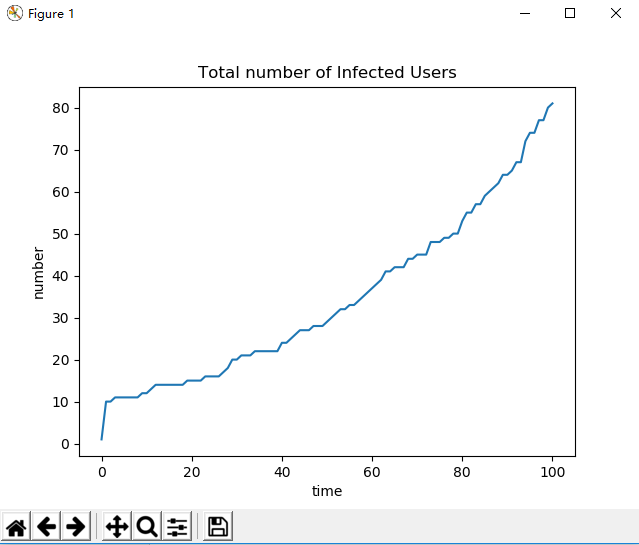
（4.1）40000个节点，节点数为M:I:S= 20079：20028：19972

感觉不太对，总数超过节点数了。



(5)SMS4.py使用28model的网络，节点数40000，10个初始感染节点，感染概率为0.0046，不考虑免疫状态，时间（循环次数）为100，感染曲线如图：

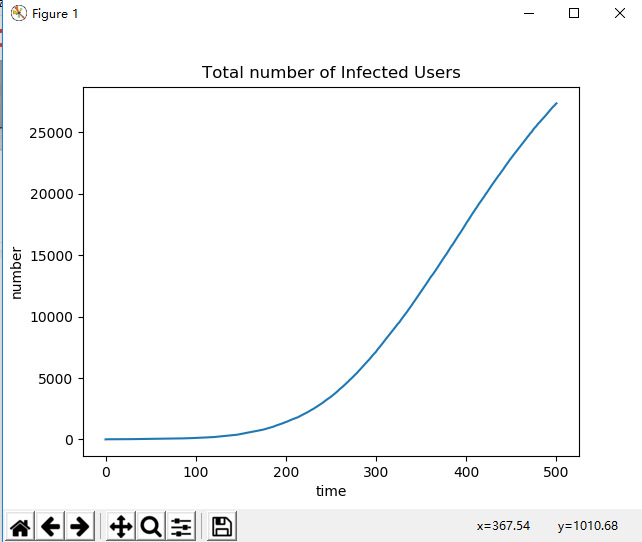
节点数为：81



Ps：节点数为4000时，感染概率0.0046时，情况同上，即循环轮次太少，

节点数40000，时间（循环次数）为500时，如图：

感染节点数27360

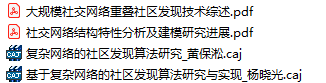


社区结构：

论文29

社区发现算法：GN算法：<https://sikasjc.github.io/2017/12/20/GN/>

4篇中文文献：



第2篇是综述型文献，对社交网络的模型特性进行分析。

在WINDOWS 操作系统，python 在本地导入文件时，地址字符串前面一定要加**r,**不让"\"做为转移字符生效。

1.对于模块和自己写的脚本不在同一个目录下，在脚本开头加sys.path.append('xxx')：

eg.　　import sys

　　　 sys.path.append(’引用模块的地址')

2.把路径添加到系统的环境变量，或把该路径的文件夹放进已经添加到系统环境变量的路径内。环境变量的内容会自动添加到模块搜索路径中。

p.s.可以通过dir(sys)来查看他里面的方法和成员属性。

<https://scikit-learn.org/stable/auto_examples/cluster/plot_digits_linkage.html#sphx-glr-auto-examples-cluster-plot-digits-linkage-py>

Different linkage type: Ward, complete, average, and single linkage

不同网络背景下这些网络的特征有什么对于传播有影响