Networks Exercise 2 In 11.18 Version 3.2

zhangtao

1 复杂网络：小世界模型

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import numpy as np  # Import modules
import networkx as nx
import matplotlib.pyplot as plt

G = nx.watts_strogatz_graph(100, 10, 0.1)  # Make graph
plt.figure(3, figsize=(12, 12), dpi=720)
nx.draw_circular(G, with_labels=False, node_size=30)
plt.savefig("G.png")
plt.show()

l_average = nx.average_shortest_path_length(G)  # Cal parame
c_average = nx.average_clustering(G)

print "average_shortest_path_length = %s" % l_average  # Pr
print "average_clustering = %s" % c_average
```

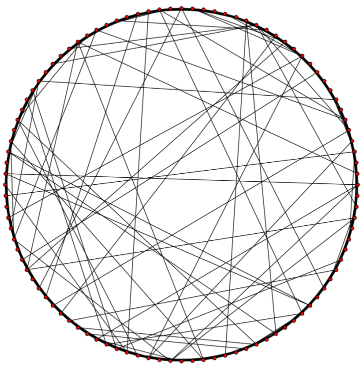

Figure 1: 小世界模型

2 画图：平均最短路径和群聚系数随概率的变化趋势

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import networkx as nx
import numpy as np
import matplotlib.pyplot as plt

l_average_list = []
c_average_list = []

p = np.linspace(0.01, 0.1, 20)

for i in range(len(p)):   # 平均化处理
    l_avg_list = []
    c_avg_list = []
    for j in range(20):
        l_average = nx.average_shortest_path_length(
            nx.watts_strogatz_graph(1000, 10, p[i]))
        c_average = nx.average_clustering(
            nx.watts_strogatz_graph(1000, 10, p[i]))
        l_avg_list.append(l_average)
        c_avg_list.append(c_average)
    l_average_list.append(float(sum(l_avg_list) / len(l_avg_list)))
    c_average_list.append(float(sum(c_avg_list) / len(c_avg_list)))

plt.title(r"$p-\bar l$")
plt.xlabel(r"$p$")
plt.ylabel(r"$\bar l$")
plt.legend()
plt.plot(p, l_average_list, label=r"$\bar l$",
         color="blue", linewidth=0, marker='o')
plt.legend()
plt.savefig("l_average.png")
plt.show()

plt.title(r"$p-\bar c$")
plt.xlabel(r"$p$")
plt.ylabel(r"$\bar c$")
plt.plot(p, c_average_list, label=r"$\bar l$",
         color="red", linewidth=0, marker='o')
plt.legend()
plt.savefig("c_average.png")
plt.show()

print "p = %s" % p
print "average_shortest_path_length = %s " % l_average_list
print "average_clustering = %s" % c_average_list
```

3 求出介数并作散点图

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Version 1.2

import itertools
import numpy as np
```

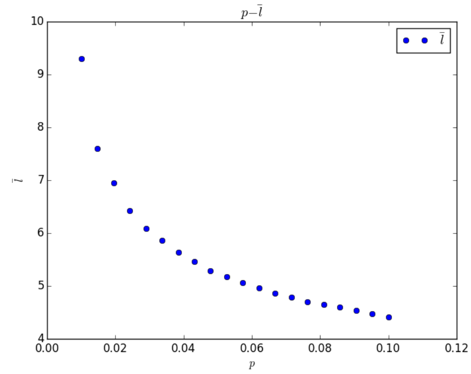

Figure 2: 平均最短路径随概率的变化

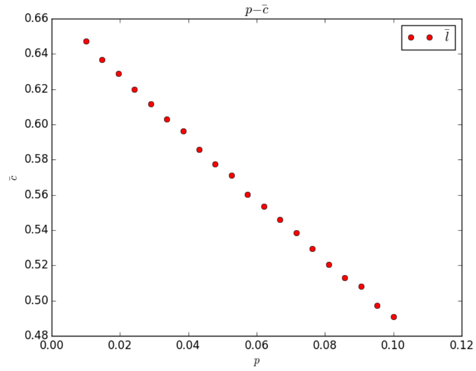

Figure 3: 群聚系数随概率的变化

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

G = nx.random_graphs.erdos_renyi_graph(100, 0.5)
degree = nx.degree_histogram(G)
k = range(len(degree))
sigma_m1 = nx.betweenness_centrality(G)
print "betweenness_centrality = %s" % sigma_m1
plt.title(r"$k- \sigma (m)$")
plt.xlabel(r"$k$")
plt.ylabel(r"$\sigma (m)$")
plt.plot(*zip(*sorted(sigma_m1.items())), color="red", linew
plt.savefig("betweenness_centrality.png")
plt.show()
```

4 画图：对平均最短路径和群聚系数作归一化处理

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
```

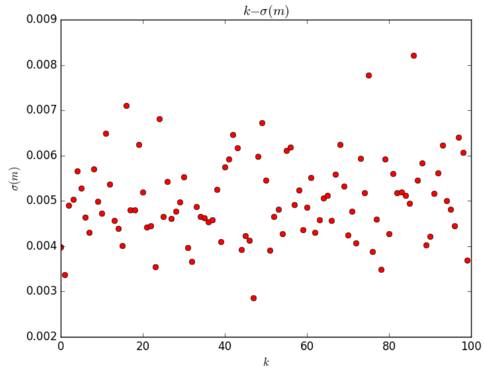

Figure 4: 介数随节点的散点图

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

l_average_list = []
c_average_list = []

l_average_max = nx.average_shortest_path_length(
    nx.watts_strogatz_graph(1000, 10, 0))
c_average_max = nx.average_clustering(nx.watts_strogatz_graph(1000, 10, 0
p = [1,
     0.00016,
     0.00028900000000000003,
     0.00049130000000000001,
     0.00083521000000000001,
     0.001419857,
     0.0024137569,
     0.0041033867299999999,
     0.0069757574409999999,
     0.011858787649699998,
     0.020159939004489997,
     0.034271896307633,
     0.0582622237229761,
     0.09904578032905938,
     0.16837782655940095,
     0.2862423051509816,
     0.48661191875666865,
     0.8272402618863367,
     1]

for i in range(len(p)):
    l_avg_list = []
    c_avg_list = []
    for j in range(20):   # 平均化处理
        l_average = nx.average_shortest_path_length(
            nx.watts_strogatz_graph(1000, 10, p[i])) / l_average_max   #
        c_average = nx.average_clustering(
            nx.watts_strogatz_graph(1000, 10, p[i])) / c_average_max
        l_avg_list.append(l_average)
        c_avg_list.append(c_average)
    l_average_list.append(float(sum(l_avg_list) / len(l_avg_list)))
    c_average_list.append(float(sum(c_avg_list) / len(c_avg_list)))

print l_average_list
print c_average_list

plt.xlabel(r"$p$")
plt.semilogx(p, l_average_list, label=r"$L(p) / L(0)$",
```

```
              color="blue", linewidth=0, marker='o')    # 对横坐标作对数处理
plt.semilogx(p, c_average_list, label=r"$C(p) / C(0)$",
              color="red", linewidth=0, marker='o')
plt.legend()  # make legend
plt.savefig("lc.png")
plt.show()
```

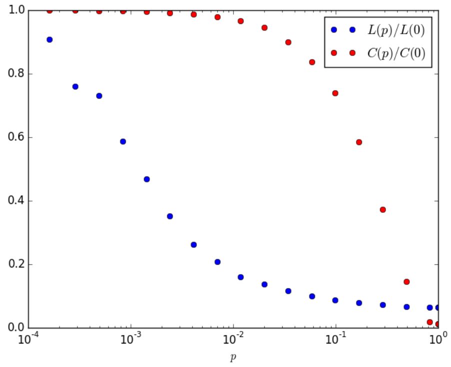

Figure 5: 平均最短路径和群聚系数随概率变化的对比图

5 小结：

通过这次作业感受到了处理数据中的几种方式：1. 平均化处理 2. 归一化处理 3. 对数坐标；以及在程序设计中的几种思维方式：1. 先数据结构的观点去看待问题 2. 从循环或者函数的角度去看待问题——循环单值循环和定义函数——可以复用 3. 算法设计及优化 … 以及小世界模型在小概率时的一些特征.