درس آمار و احتمال مهندسی

- گزارش پروژه

متاگراف

تهیه کنندگان: آرمان لطفعلی خانی (۹۹۱۰۹۱۶۶)

محمدامین انصاری جعفری (۹۹۱۰۱۲۳۵)

استاد درس: دکتر مداح علی

تاریخ: ۱۴۰۰/۱۱/۱۴

دانشکده مهندسی برق

فهرست مطالب

[۱] درخت دوستی بنشان که کام دل به بار آرد/ نهال دشمنی برکن که رنج بی شمار آرد [حافظ]

۱ درخت دوستی بنشان! [۲]

۱.۱ پاسخ پرسش تئوری اول

با توجه به اینکه احتمال وجود هر یال p و m یال هم از کل یال ها در جای خود باید درست تعیین شوند، داریم:

$$\mathbb{P}(G=g) = p^m(1-p)^{\binom{n}{2}-m} \tag{1}$$

۲.۱ پاسخ پرسش تئوری دوم

از کل یال ها باید از m یالی که انتخاب کرده است یک حالت درست را تشخیص دهد، پس:

$$\mathbb{P}(G=g) = \frac{1}{\binom{\binom{n}{2}}{m}} \tag{2}$$

۳.۱ پاسخ پرسش تئوری سوم

یال‌های اضافی خارج از مجموعه دوستی‌های واقعی برای ما اهمیت ندارد، پس کافیست یال‌های مربوط به m دوستی را بررسی کنیم. اگر متغیر تصادفی X را درصد انتخاب درست هکر در نظر بگیریم، احتمال گفته شده به صورت زیر خواهد بود:(ممکن است ۵ m را عاد نکند پس ما ۲۰ یا اندکی بیش از آن را در صورت سوال در نظر می‌گیریم)

$$\mathbb{P}(X \approx) = \binom{m}{\lceil \frac{m}{5} \rceil} p^{\lceil \frac{m}{5} \rceil}(1-p)^{m-\lceil \frac{m}{5} \rceil} \tag{3}$$

با تو

۴.۱ شبیه سازی اول

```python
# Part 1

import networkx as nx

n = 1000
repeat = 10
p = 0.0034

s = 0

for i in range(repeat):
    G = nx.fast_gnp_random_graph(n,p)
    s += G.number_of_edges()

print(s/repeat)
```

برنامهٔ ۱: کد شبیه‌سازی اول

[۲] درخت دوستی بنشان که کام دل به بار آرد/ نهال دشمنی برکن که رنج بی شمار آرد [حافظ]

1701.6

شکل ۱: نمونه خروجی شبیه‌سازی

برای این شبیه سازی از دستور آماده کتاب‌خانه ی networkx برای یافتن تعداد یال ها و ساخت گراف تصادفی استفاده میکنیم و آن را ۱۰ بار تکرار می‌کنیم. خروجی با m ای که صورت سوال داده است تفاوت زیادی دارد اما همانطور که در تئوری بعدی اثبات میکنم، به صورت تئوری به شکل زیر است:

$$m = ۰/۰۰۳۴ \times \binom{۱۰۰۰}{۲} = ۱۶۹۸/۳ \tag{۴}$$

که با شبیه سازی تطابق دارد.

۵.۱ پاسخ پرسش تئوری چهارم

با توجه به دانستن تعداد کل یال ها و احتمال هر کدام، فرض می‌کنیم X متغیر تصادفی تعداد یال ها است و داریم:

$$\mathbb{E}[X] = p\binom{n}{۲} \tag{۵}$$

پس برای برابری تقریبی با m باید داشته باشیم:

$$m \simeq p\binom{n}{۲} \tag{۶}$$

۲ خلوت گزیده را به تماشا چه حاجت است؟

۱.۲ شبیه سازی دوم

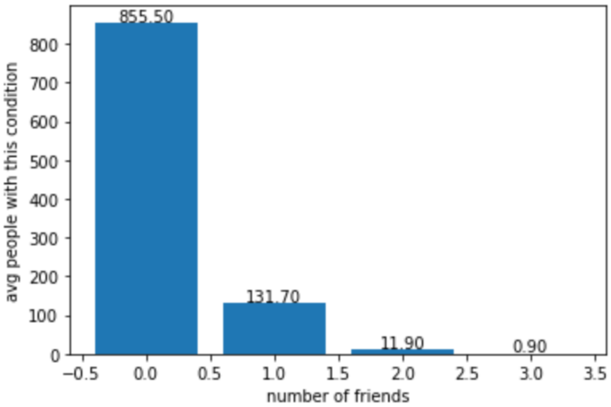
```python
# Part 2

import networkx as nx
import matplotlib.pyplot as plt

n = 1000
p = 0.00016
repeat = 10

Glist = []
sprime = 0


for j in range(repeat)
    G = nx.fast_gnp_random_graph(n,p)
    Glist.append(G)

    s = 0
```

```
    for i in range(n)
        s = s + G.degree[i]
    L = s/n

    for i in range(n)
        if  G.degree[i] = L
            sprime = sprime + 1
print(sprimerepeat)

flist = []
xlist = []
s = 0

for i in range(n)
    for G in Glist
        for j in range(n)
            if G.degree[j] == i
                s = s + 1
    if s != 0
       flist.append(srepeat)
       xlist.append(i)
    s = 0

plt.bar(xlist,flist)

for x,y in zip(xlist,flist)

    label = {.2f}.format(y)

    plt.annotate(label,
    (x,y),
    textcoords=offset points,
    xytext=(0,1),
    ha='center')

plt.xlabel('number of friends')

plt.ylabel('avg people with this condition')

plt.title('')

plt.show()
```

برنامهٔ ۲: کد شبیه‌سازی دوم

144.5

شکل ۲: نمونه خروجی شبیه‌سازی



شکل ۳: نمودار خروجی شبیه‌سازی

ابتدا به محاسبه ی L می پردازیم و سپس متوسط افراد اجتماعی را میابیم. در نمودار هم برای نمایش بهتر، تعداد افرادی که صفر بودند، حذف شده است و در دیگر حالات برای تعداد دوست ها متوسط این افراد صفر است.

۲.۲ پاسخ پرسش تئوری پنجم

هر نفر با $n-۱$ میتواند تشکیل دوستی دهد که هر کدام احتمال p دارد، پس:

$$\mathbb{E}[X] = p(n-۱) \tag{۷}$$

و با اعداد گفته شده در سوال داریم:

$$\mathbb{E}[X] = ۰/۰۰۰۱۶ \times (۱۰۰۰ - ۱) = ۰/۱۵۹۸۴ \tag{۸}$$

۳.۲ پاسخ پرسش تئوری ششم

تعداد دوستان راس i ام را X_i و مجموع تعداد دوستان هر شخص را X در نظر می‌گیریم. مشخصا رابطه $\sum_{i=۱}^{\binom{n}{۲}} X_i$ برقرار است. در ابتدا برای احتمال اجتماعی بودن فرد مورد نظر داریم:

$$\mathbb{P}(X_i > L) = ۱ - \mathbb{P}(X_i < L) = ۱ - \mathbb{P}(X_i < ۰/۱۵۹۸۴) = ۱ - \mathbb{P}(X_i = ۰) = ۱ - (۱-p)^{n-۱} \tag{۹}$$

سپس بر اساس خاصیت خطی بودن امید ریاضی داریم:

$$\mathbb{E}[X] = \sum_{i=۱}^{\binom{n}{۲}} \mathbb{E}[X_i] = n(۱ - (۱-p)^{n-۱}) \simeq ۱۴۷/۷۳۰۷۵۶ \tag{۱۰}$$

۳ هواداران کویش را چو جان خویشتن دارم؟

۱.۳ شبیه سازی سوم

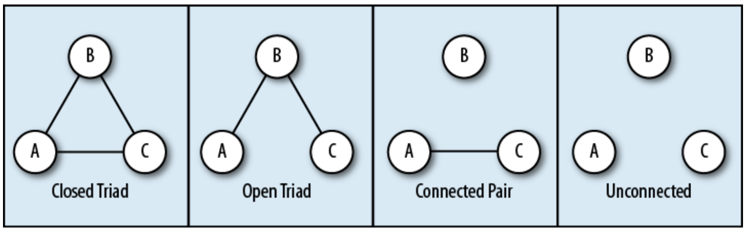
```python
# Part 3

import networkx as nx

n = 3000
p = 0.01
repeat = 5

number_of_triangles = 0
number_of_contests = 0

for m in range(repeat):

    G = nx.fast_gnp_random_graph(n,p)

    H = G.to_directed()
    T = nx.triadic_census(H)

    if '300' in T.keys():
        number_of_triangles = number_of_triangles + T.get('300')
    if '201' in T.keys():
        number_of_contests = number_of_contests + T.get('201')

print(number_of_triangles/repeat)
print(number_of_contests/repeat)
```

برنامهٔ ۳: کد شبیه‌سازی سوم

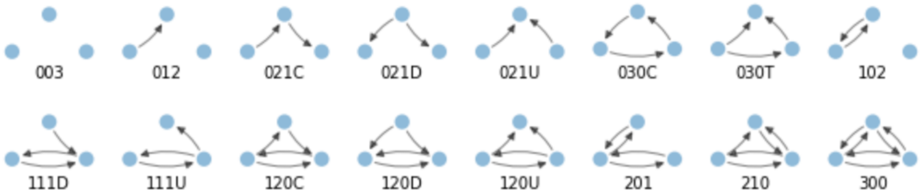
```
print(number_of_triangles/repeat)
print(number_of_contests/repeat)

4492.0
1329365.8
```

شکل ۴: نمونه خروجی شبیه‌سازی

برای انجام این شبیه سازی کتاب خانه ی networkx دستور آماده ای برای گراف بدون جهت ندارد اما این کتاب خانه برای گراف های جهت دار دستور های جالب و کاربردی دارد پس ما گراف را به گراف جهت دار (هر یال را به دو یال جهت دار در خلاف جهت هم) تبدیل میکنیم. دستور آماده ی `nx.triadic_census(G)` لیستی از تعداد حالت های ممکن که در شکل زیر مشخص شده اند، به ما می دهد و به راحتی حالت های ممکن خود را انتخاب میکنیم.



شکل ۵: حالت های سه گره ای برای گراف بدون جهت



شکل ۶: حالت های سه گره ای برای گراف جهت دار

شاید خوب باشد که اشاره کنیم اگر از این دستور استفاده نکنیم زمان اجرای کد ممکن است از حدوداً یک دقیقه ای الان به حدود ۱۰ دقیقه تبدیل بشود!

۲.۳ شبیه سازی چهارم

```
# Part 4

import networkx as nx

n = 2000
p = 0.2
repeat = 3

number_of_triangles = 0
number_of_contests = 0

for m in range(repeat):

    G = nx.fast_gnp_random_graph(n,p)

    H = G.to_directed()

    print("round "+str(m+1)+" started!")

    T = nx.triadic_census(H)

```

```python
        if '300' in T.keys():
            number_of_triangles = number_of_triangles + T.get('300')
        if '201' in T.keys():
            number_of_contests = number_of_contests + T.get('201')

        print("round "+str(m+1)+" completed!")

print(number_of_triangles/repeat)
print(number_of_contests/repeat)
```

برنامهٔ ۴: کد شبیه‌سازی چهارم

```
print(number_of_triangles/repeat)
print(number_of_contests/repeat)
```

```
round 1 started!
round 1 completed!
round 2 started!
round 2 completed!
round 3 started!
round 3 completed!
10666854.666666666
127907403.33333333
```

شکل ۷: نمونه خروجی شبیه‌سازی

دقیقا مانند شبیه سازی قبل انجام میدهیم و تغییرات مورد نظر در فرضیات را اعمال میکنیم. در این شبیه سازی زمان اجرای کد به علت بیشتر شدن تعداد یال ها، خیلی بیشتر خواهد بود.

۳.۳ پاسخ پرسش تئوری هفتم

با توجه به فرضیات و دانش ترکیباتی خود، برای تراگذری سه یال داریم که باید سه گره از گره ها هم انتخاب کنیم و برای رقابت سه گره داریم و دو یال که با انتخاب دو از ۳ جایگشت نیز دارند، پس:

$$\begin{aligned} \mathbb{E}[\text{تراگذری}] &= p^{۳}\binom{n}{۳} \\ \mathbb{E}[\text{رقابت}] &= p^{۲}(۱-p)\binom{n}{۳}\binom{۳}{۲} = ۳p^{۲}(۱-p)\binom{n}{۳} \end{aligned} \tag{۱۱}$$

۴.۳ پاسخ پرسش تئوری هشتم

با استفاده از معادله ۱۱ داریم:

$$\frac{p^{۳}\binom{n}{۳}}{p^{۳}\binom{n}{۳} + ۳p^{۲}(۱-p)\binom{n}{۳}} = \frac{p}{۳-۲p} \tag{۱۲}$$

برای بررسی با شبیه سازی داریم:

$$\begin{aligned}\mathbb{E}[\text{تراگذری}] &= 0.01^3 \times \binom{3000}{3} = 4495.501 \\ \mathbb{E}[\text{رقابت}] &= 3 \times 0.01^2 \times (1 - 0.01)\binom{3000}{3} = 1335163.797\end{aligned} \tag{13}$$

برای شبیه سازی دوم هم:

$$\begin{aligned}\mathbb{E}[\text{تراگذری}] &= 0.2^3 \times \binom{2000}{3} = 10650672 \\ \mathbb{E}[\text{رقابت}] &= 3 \times 0.2^2 \times (1 - 0.2)\binom{2000}{3} = 127808064\end{aligned} \tag{14}$$

که به صورت تقریبی برابر اعداد شبیه سازی بوده و با آن‌ها تطابق دارند.

۵.۳ شبیه سازی پنجم

```python
# Part 5

import random
import networkx as nx

n = 1000
p = 0.003

G = nx.fast_gnp_random_graph(n,p)

s = 0

for i in range(n):
    s += len(list(G.subgraph(G.neighbors(i)).edges))

print(s/n)
```

برنامهٔ ۵: کد شبیه‌سازی پنجم

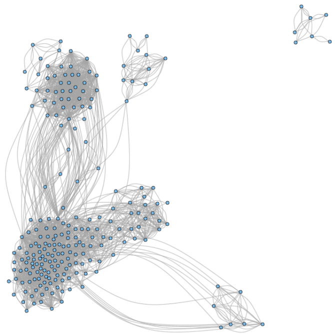
```
0.012
```

شکل ۸: نمونه خروجی شبیه‌سازی

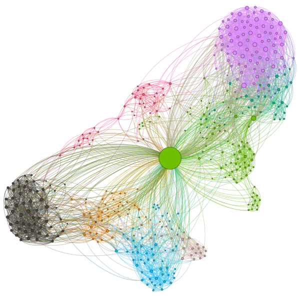
برای این شبیه سازی از دستورات مربوط به زیر گراف و گره های مجاور استفاده کردیم.

۶.۳ پاسخ پرسش تئوری نهم

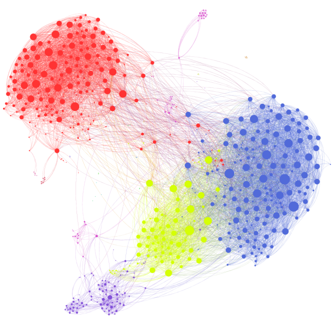
به علاوه ی تصویر داخل صورت سوال نظرتان را به چند نمونه دیگر جلب میکنیم:



شکل ۹: نمونه اول گراف شبکه اجتماعی فیسبوک



شکل ۱۰: نمونه دوم گراف شبکه اجتماعی فیسبوک



شکل ۱۱: نمونه سوم گراف شبکه اجتماعی فیسبوک

همانطور که می‌بینیم، بر خلاف گراف تصادفی که به صورت کاملا تصادفی یال ها وجود داشتند، در اینجا تجمع یال ها در همسایگی گره ها بسیار بیشتر است تا حدی که این قسمت ها شلوغ به زیر گراف تقریب کامل تبدیل می شوند! پس نتیجه در واقعیت بیشتر از نتیجه ی شبیه سازی خواهد بود.

۷.۳ پاسخ پرسش تئوری دهم

در واقع ما به دنبال تراگذری هستیم که از طرفی به طور تصادفی دنبال دو گره به جز شخص مد نظر هستیم پس با توجه به فرضیات داریم:

$$\mathbb{E}[X] = p^{3}\binom{n-1}{2} \qquad (15)$$

۴ من از دیار حبیبم نه از بلاد غریب!

۱.۴ شبیه سازی ششم

```python
# Part 6

import networkx as nx

n = 1000
p = 0.0033

G = nx.fast_gnp_random_graph(n,p)
s = 0

for i in range(n):
    for j in range(1,n):
        if nx.has_path(G, i, j):
            s = s + nx.shortest_path_length(G,i,j)

print(s/(n*(n-1)/2))
```

برنامهٔ ۶: کد شبیه‌سازی ششم

```
10.481387387387388
```

شکل ۱۲: نمونه خروجی شبیه‌سازی

در این شبیه سازی از دستور آماده کوتاه ترین مسیر را محاسبه میکنیم.

۲.۴ شبیه سازی هفتم

```python
# Part 7

import networkx as nx

n = 50
p = 0.34
repeat = 100

s = 0

for i in range(repeat):
    G = nx.fast_gnp_random_graph(n,p)

    s += nx.diameter(G)

print(s/repeat)
```

برنامهٔ ۷: کد شبیه‌سازی هفتم

```
2.77
```

شکل ۱۳: نمونه خروجی شبیه‌سازی

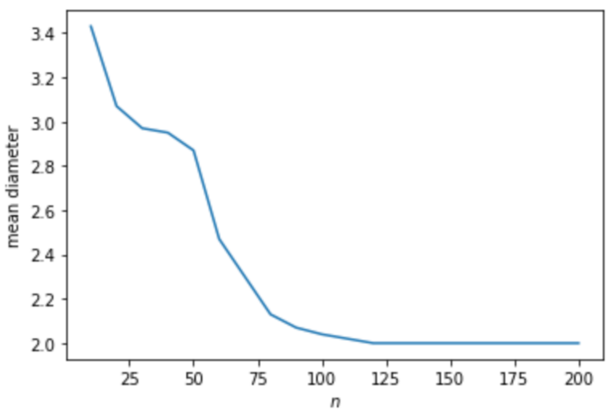
با توجه به تابع آماده برای بدست آوردن قطر گراف، قطر گراف را در حالت های مختلف بدس می آوریم و سپس میانگین می گیریم.

۳.۴ شبیه سازی هشتم

```python
# Part 8

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

a = 10
b = 200
step = 10
n = np.arange(a, b + step, step).tolist()
p = 0.34
repeat = 100

s = 0
diaList = []
```

```

for nprime in  n:

    for j in range(repeat):
        G = nx.fast_gnp_random_graph(nprime,p)

        if not nx.is_connected(G):
            maxL = 0
            for Gprime in nx.connected_components(G):
                diag = nx.diameter(G.subgraph(Gprime))
                if diag > maxL:
                   maxL = diag
            s+=maxL
        else :
            s += nx.diameter(G)

    diaList.append(s/repeat)
    s = 0

plt.plot(n,diaList)
plt.ylabel('mean diameter')
plt.xlabel('$n$')
plt.show()
```

برنامهٔ ۸: کد شبیه‌سازی هشتم



شکل ۱۴: نمودار خروجی شبیه‌سازی

مانند قسمت قبل عمل کنیم اما چون با کم شدن یال ها احتمال تشکیل گراف جدا میشود، این حالت را هم در کد مد نظر قرار می دهیم. همانطور که مشاهده می شود، مقدار نمودار زمانی که n به بی‌نهایت می‌رود، به ۲ میل میکند. نمودار هم نزولی اکید است و مجانب در مقدار ۲ دارد.

۴.۴ پاسخ پرسش تئوری یازدهم

برای نداشتن همسایه مشترک، باید هر گره ای به جز این دو تا دارای دو یال متصل از این دو گره به خودش نباشد، پس: (p^2 احتمال داشتن هر دو یال با یک گره توسط این دو گره است)

$$\mathbb{P}(I_{u,v} = 1) = (1 - p^2)^{n-2} \tag{16}$$

۵.۴ پاسخ پرسش تئوری دوازدهم

با توجه به خاصیت خطی امید ریاضی و شمارش دوتایی های ممکن با $\binom{n}{2}$ داریم:

$$\mathbb{E}[X_n] = \sum_{u,v \in n} \mathbb{P}(I_{u,v} = 1) = \binom{n}{2}(1 - p^2)^{n-2} \tag{17}$$

۶.۴ پاسخ پرسش تئوری سیزدهم

به علت سرعت رشد تابع نمایی که بیشتر از توان چندجمله‌ای است و پایه این رشد نمایی کوچکتر از ۱ است پس امید ریاضی که در قسمت قبل بدست آوردیم، در n های بسیار بزرگ به صفر میل می‌کند:

$$\begin{aligned} &\mathbb{P}(1 \leqslant X_n) \leqslant \mathbb{E}[X_n] \\ &\xrightarrow{n \to \infty} \mathbb{P}(1 \leqslant X_n) \to 0 \end{aligned} \tag{18}$$

۷.۴ پاسخ پرسش تئوری چهاردهم

طبق قسمت قبل در n های خیلی بزرگ، هر دو راس حتما همسایه مشترک دارند پس قطر گراف به ۲ میل میکند که این کران بالای قطر گراف می شود. در این صورت با ثابت بودن p، دیگر قطر گراف به p وابسته نیست. این نتیجه با مشاهدات از شبیه سازی کاملا همخوانی دارد.

۵ وَ إن یَکاد بخوانید!

۱.۵ شبیه سازی نهم

کافیست که از دستور triangles برای شمارش مثلث‌های شامل هر گره استفاده کرده سپس تمام این اعداد را با هم جمع کنیم. با توجه به اینکه در هر مثلث، ۳ راس وجود دارند، نتیجه نهایی باید تقسیم بر ۳ شود.

```python
import numpy as np
import matplotlib.pyplot as mplot
from numpy.random import default_rng
import networkx as nx

def mean_triangles(n,p,iteration):
    c=0
    for i in range(iteration):
        g=nx.fast_gnp_random_graph(n,p)
        nodes=g.nodes()
        for node in nodes:
            c=c+nx.triangles(g,node)
    return c/(3*iteration)
```

```
print(mean_triangles(100,0.34,100))
```

برنامهٔ ۹: کد شبیه‌سازی نهم

نتیجه شبیه‌سازی در شکل ۱۵ قابل مشاهده است:

```
import numpy as np
import matplotlib.pyplot as mplot
from numpy.random import default_rng
import networkx as nx

def mean_triangles(n,p,iteration):
    c=0
    for i in range(iteration):
        g=nx.fast_gnp_random_graph(n,p)
        nodes=g.nodes()
        for node in nodes:
            c=c+nx.triangles(g,node)
    return c/(3*iteration)
print(mean_triangles(100,0.34,100))
```

6336.8

شکل ۱۵: کد و خروجی شبیه‌سازی

همان‌طور که مشاهده می‌شود، عدد بدست آمده با عددی که از تئوری بدست می آید (که رابطه آن در بخش‌های تئوری بعدی اثبات خواهد شد) همخوانی دارد.

$$\binom{۱۰۰}{۳} \times (۰/۳۴)^{۳} \approx ۶۳۵۵$$

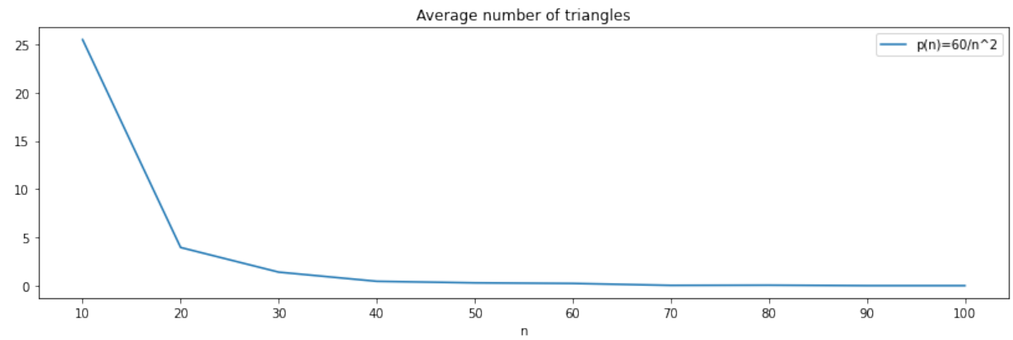
۲.۵ شبیه سازی دهم

در کد این بخش، تنها لازم است که تابع قسمت قبل از vectorize کنیم تا برای کار با آرایه مناسب شود.ضمنا آرایه p را طبق رابطه خواسته شده تشکیل می‌دهیم و از آن برای ورودی دادن به تابع استفاده میکنیم. در پایان، خروجی‌های تابع را برای رسم نمودار استفاده می‌نماییم.

```
import numpy as np
import matplotlib.pyplot as mplot
from numpy.random import default_rng
import networkx as nx
@np.vectorize
def mean_triangles(n,p,iteration):
    c=0
    for i in range(iteration):
        g=nx.fast_gnp_random_graph(n,p)
        nodes=g.nodes()
        for node in nodes:
            c=c+nx.triangles(g,node)
    return c/(3*iteration)
```

```
x=np.arange(10,110,10)
p=60/x**2
mean=mean_triangles(x,p,100)
fig1,ax1=mplot.subplots(figsize=(14, 4))
mplot.xticks(np.arange(0,110,10))#suitable value ticks on x axis
ax1.set_title('Average number of triangles')
mplot.xlabel('n')
mplot.plot(x,mean,label="p(n)=60/n^2")
mplot.legend()
```

برنامهٔ ۱۰: کد شبیه‌سازی دهم



شکل ۱۶: نمودار میانگین تعداد مثلث‌ها برحسب تعداد راس‌ها

مشاهده می‌شود که میانگین با افزایش تعداد راس‌ها به صفر میل می‌کند. علت این اتفاق نیز کاهش بسیار سریع p می‌باشد به طوری که در n زیاد عملا یال‌های بسیار کمی داریم (از رابطه تئوری امیدریاضی تعداد یال‌ها استفاده کنیم به $\frac{1}{2}$ می‌رسیم که از ۱ نیز کمتر است). این نتیجه نیز به صورت تئوری در بخش‌های بعدی اثبات خواهد شد.

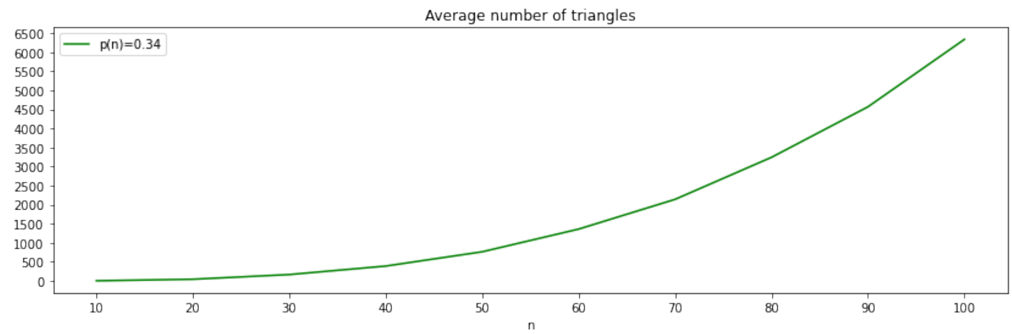
۳.۵ شبیه سازی یازدهم

کد تغییرات بسیار اندکی نسبت به بخش پیشین دارد. صرفا رابطه $p(n)$ عوض شده و یک خط کد برای تنظیم علامت‌گذاری‌های نمودار اضافه شده است.

```
import numpy as np
import matplotlib.pyplot as mplot
from numpy.random import default_rng
import networkx as nx
@np.vectorize
def mean_triangles(n,p,iteration):
    c=0
    for i in range(iteration):
        g=nx.fast_gnp_random_graph(n,p)
        nodes=g.nodes()
        for node in nodes:
            c=c+nx.triangles(g,node)
    return c/(3*iteration)
x=np.arange(10,110,10)
```

```
p=0.34
mean=mean_triangles(x,p,100)
fig1,ax1=mplot.subplots(figsize=(14, 4))
ax1.set_title('Average number of triangles')
mplot.xticks(np.arange(10,110,10))#suitable value ticks on x axis
mplot.yticks(np.arange(0,6600,500))#suitable value ticks on y axis
mplot.xlabel('n')
mplot.plot(x,mean,'g',label="p(n)=0.34")
mplot.legend()
```

برنامهٔ ۱۱: کد شبیه‌سازی یازدهم



شکل ۱۷: نمودار میانگین تعداد مثلث‌ها برحسب تعداد راس‌ها به ازای رابطه p متفاوت

همان طور که انتظار می‌رود، میانگین تعداد مثلث‌ها کاملا سیر صعودی و رو به افزایش دارد و به عدد خاصی میل نمی‌کند. به طور کیفی نیز می‌توان گفت که احتمال تشکیل هر مثلث ثابت مانده ولی تعداد راس‌ها و در نتیجه حداکثر تعداد مثلث‌های ممکن افزایش یافته است. رابطه تئوری نیز همین را به ما می‌گوید:

$$\lim_{n\to\infty} \binom{n}{۳} c^{۳} = \infty$$

۴.۵ شبیه سازی دوازدهم

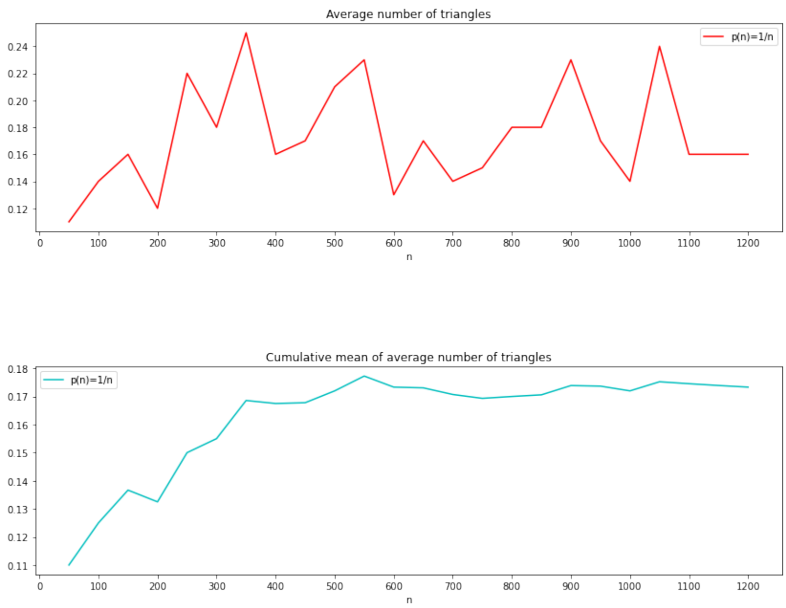
کد همچنان از لحاظ توابع تعریف شده و چگونگی استفاده از آنها، تفاوت چندانی با بخش‌های قبلی ندارد. بازه‌بندی در خط ۱۴ تغییر یافته و در خطوط ۱۷ تا ۲۱، با یک حلقه و ذخیره کردن مجموع i+1 خانه اول، آرایه میانگین تجمعی نمونه‌ها تشکیل داده شده است. خطوط بعدی نیز مشابه قبل بوده و برای رسم نمودارها و تنظیم جزئیات آنها می‌باشند.

```
۱  import numpy as np
۲  import matplotlib.pyplot as mplot
۳  from numpy.random import default_rng
۴  import networkx as nx
۵  @np.vectorize
۶  def mean_triangles(n,p,iteration):
۷      c=0
۸      for i in range(iteration):
۹          g=nx.fast_gnp_random_graph(n,p)
۱۰         nodes=g.nodes()
۱۱         for node in nodes:
```

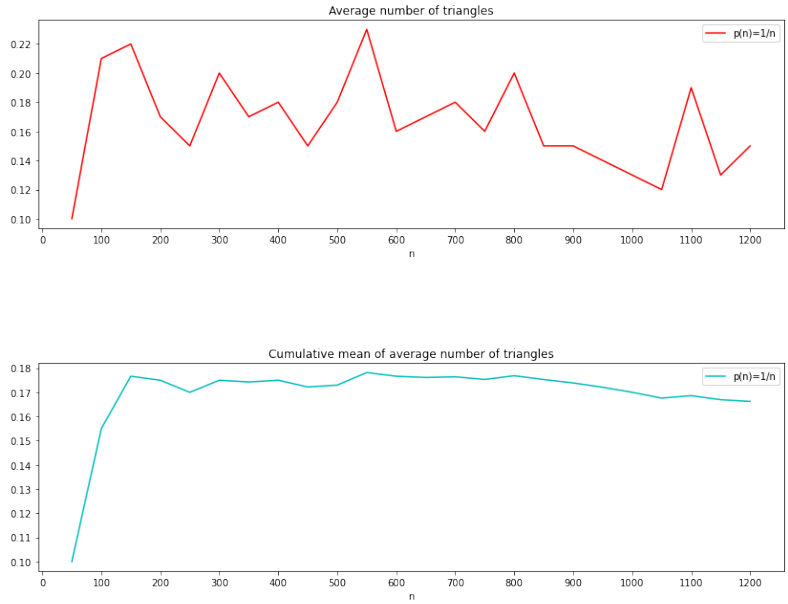
```
            c=c+nx.triangles(g,node)
    return c/(3*iteration)
x=np.arange(50,1250,50)
p=1/x
mean=mean_triangles(x,p,100)
cumsum=0
cummean=np.zeros(x.size)
for i in range(x.size):
    cumsum=cumsum+mean[i]
    cummean[i]=cumsum/(i+1)
fig1,ax1=mplot.subplots(figsize=(14, 4))
ax1.set_title('Average number of triangles')
mplot.xlabel('n')
mplot.xticks(np.arange(0,1300,100))#suitable value ticks on x axis
mplot.plot(x,mean,'r',label="p(n)=1/n")
mplot.legend()
fig2,ax2=mplot.subplots(figsize=(14, 4))
mplot.xticks(np.arange(0,1300,100))#suitable value ticks on x axis
ax2.set_title('Cumulative mean of average number of triangles')
mplot.xlabel('n')
mplot.plot(x,cummean,'c',label="p(n)=1/n")
mplot.legend()
```

برنامهٔ ۱۲: کد شبیه‌سازی دوازدهم

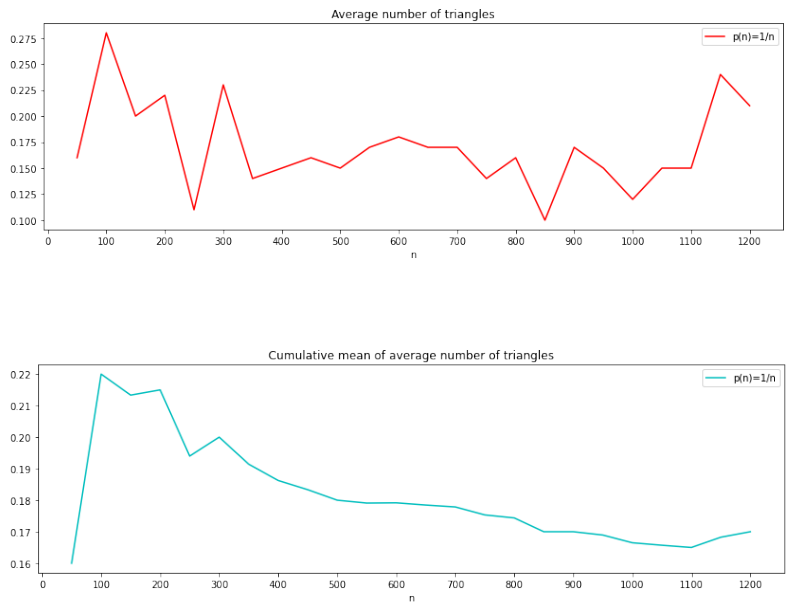
خروجی کد به ازای سه بار ران کردن، در شکل‌های ?? تا ?? قابل مشاهده می‌باشد.



شکل ۱۸: نمودارهای میانگین و میانگین تجمعی تعداد مثلث‌ها (بار اول)



شکل ۱۹: نمودارهای میانگین و میانگین تجمعی تعداد مثلث‌ها (بار دوم)



شکل ۲۰: نمودارهای میانگین و میانگین تجمعی تعداد مثلث‌ها (بار سوم)

همان طور که مشاهده می‌شود، نه میانگین تعداد مثلث‌ها و نه شکل کلی نمودار میانگین تجمعی رفتار یکسان و ثبل پیش‌بینی ندارند، ولی چند خصوصیت کلی از این نمودارها قابل استخراج است. از تئوری می‌دانیم که به ازای $p(n(=\frac{c}{n}$ و $n \gg ۱$ رابطه

زیر برای میانگین تعداد مثلث‌ها برقرار است:

$$\mathbb{E}(T_{3,n}) = \binom{n}{3} p^3 \approx \frac{c^3}{6} \xRightarrow{c=1} \mathbb{E}(T_{3,n}) = \frac{1}{6} \approx 0/167 \qquad (19)$$

مشاهده می‌شود که میانگین تعداد مثلث‌ها به طور تقریبی حول همین مقدار نوسان می‌کند. همچنین، نقطه پایانی نمودارهای میانگین تجمعی که برابر میانگین کل خروجی‌ها می‌باشد، در هر سه نمودار با وجود رفتارهای متفاوت در هر نمودار، ثابت و نزدیک به همین مقدار تئوری می‌باشد. همچنین، خروجی‌های اول میانگین تجمعی به علت تعداد کم در میانگین‌گیری، قابل اتکا نیستند و دلیلی برای خوش رفتاری آن‌ها نداریم. طبیعتا با افزایش تعداد نمونه‌ها نتایج بیش از پیش به مقدار تئوری نزدیک خواهند شد.

۵.۵ پاسخ پرسش تئوری پانزدهم

با توجه به مستقل بودن احتمال تشکیل شدن یال‌ها از همدیگر، داریم:

$$\mathbb{P}(I_{u,v,w} = 1) = \begin{cases} p^3 & u \neq v \neq w \\ 0 & o.w \end{cases} \qquad (20)$$

۶.۵ پاسخ پرسش تئوری شانزدهم

با توجه به قسمت قبل داریم:

$$\begin{aligned} \mathbb{E}(T_{3,n}) &= \sum_{u,v,w} \mathbb{P}(I_{u,v,w} = 1) \times 1 + \mathbb{P}(I_{u,v,w} = 0) \times 0 = \sum_{\substack{u,v,w \\ u\neq v\neq w}} \mathbb{P}(I_{u,v,w} = 1) \\ &= \sum_{\substack{u,v,w \\ u\neq v\neq w}} p^3 = \binom{n}{3} p^3 \end{aligned} \qquad (21)$$

در ساده سازی عبارت آخر، از این که مقدار سری برابر با تعداد کل مثلث‌های ممکن مدنظر است، استفاده کردیم.

۷.۵ پاسخ پرسش تئوری هفدهم

$$\begin{aligned} & 0 \leq \mathbb{P}(T_{3,n} \geq 1) \leq \frac{\mathbb{E}(T_{3,n})}{1} \Rightarrow 0 \leq P(T_{3,n} \geq 1) \leq p^3 \binom{n}{3} \\ & \binom{n}{3} = \frac{n(n-1)(n-2}{6} \leq \frac{n^3}{6}, p(n) = \frac{1}{n^2} \Rightarrow 0 \leq P(T_{3,n} \geq 1) \leq \frac{1}{n^6}\frac{n^3}{6} = \frac{1}{6n^3} \\ & \lim_{n\to\infty} \frac{1}{6n^3} = 0 \Rightarrow \lim_{n\to\infty} P(T_{3,n} \geq 1) = 0 \end{aligned} \qquad (22)$$

مشاهده می‌شود که با افزایش تعداد راس‌ها، احتمال تشکیل مثلث به صفر میل می‌کند.

۸.۵ پاسخ پرسش تئوری هجدهم

با استفاده از نامساوی مارکف (یا چبیشف)، نامنفی و گسسته بودن متغیر تصادفی و قضیه فشار در حد، می‌توان روابط زیر را نوشت:

$$\begin{aligned}
&\mathbb{P}(|Y_n-\mathbb{E}(Y_n)|\geq \mathbb{E}(Y_n))=\mathbb{P}(|X-\mathbb{E}(Y_n)|^۲\geq \mathbb{E}(Y_n)^۲)\leq \frac{Var(Y_n)}{\mathbb{E}(Y_n)^۲}\\
&Y_n\geq ۰\Rightarrow \mathbb{P}(|Y_n-\mathbb{E}(Y_n)|\geq \mathbb{E}(Y_n))=\mathbb{P}(Y_n=۰)+\mathbb{P}(Y_n\geq ۲\mathbb{E}(Y_n))\geq \mathbb{P}(Y_n=۰)\\
&\Rightarrow ۰\leq \mathbb{P}(Y_n=۰)\leq \frac{Var(Y_n)}{\mathbb{E}(Y_n)^۲}\\
&\lim_{n\to\infty}\frac{Var(Y_n)}{\mathbb{E}(Y_n)^۲}=۰\Rightarrow \lim_{n\to\infty}\mathbb{P}(Y_n=۰)=۰ \qquad \text{قضیه فشار}\\
&\mathbb{P}(Y_n<۱)=\mathbb{P}(Y_n=۰)=۱-\mathbb{P}(Y_n\geq ۱)\Rightarrow \lim_{n\to\infty}\mathbb{P}(Y_n\geq ۱)=۱ \quad \text{گسسته بودن متغیر تصادفی}
\end{aligned} \tag{۲۳}$$

۹.۵ پاسخ پرسش تئوری نوزدهم

$$\mathbb{E}(I_{u',v',w'}I_{u,v,w})=\mathbb{P}(I_{u',v',w'}=۱)\mathbb{P}(I_{u,v,w}=۱|I_{u',v',w'}=۱)=p^۳\mathbb{P}(I_{u,v,w}=۱|I_{u',v',w'}=۱) \tag{۲۴}$$

حال با حالت بندی روی تعداد اشتراکات اندیس‌های دو متغیر تصادفی و در نتیجه تعداد اضلاع مشترک دو مثلث، با فرض $u\neq v\neq w$ و $u'\neq v'\neq w'$ داریم:

$$\mathbb{P}(I_{u,v,w}=۱|I_{u',v',w'}=۱)=\begin{cases}
۱ & u=u',v=v',w=w'\\
p^۳ & u=u',v\neq v',w\neq w'\\
 & u\neq u',v=v',w\neq w'\\
 & u\neq u',v\neq v',w\neq w'\\
p^۳ & u\neq u',v\neq v',w\neq w'\\
p^۲ & u=u',v=v',w\neq w'\\
 & u=u',v\neq v',w=w'\\
 & u=u',v=v',w\neq w'
\end{cases} \tag{۲۵}$$

۱۰.۵ پاسخ پرسش تئوری بیستم

برای محاسبه این عبارت، ابتدا باید تعداد ظاهر شدن هر یک از حالت‌های ذکر شده در بخش قبلی را به ازای u', v', w' مشخص، بدست آوریم.

$$\begin{aligned}
&\text{تعداد حالت‌های یکی بودن هر سه اندیس (ردیف اول حالت بندی)}=۱\\
&\binom{۳}{۱}\binom{n-۳}{۲}=\text{تعداد حالاتی که یک راس از مثلث اول و دوم مشترک باشند}\\
&\binom{n-۳}{۳}=\text{تعداد حالاتی که هیچ یک از ۳ راس دوم با قبلی‌ها اشتراک نداشته باشند}\\
&\binom{n-۳}{۱}\binom{۳}{۲}=\text{تعداد حالاتی که ۲ راس از راس مثلث دوم، با مثلث اول مشترک باشنذ}
\end{aligned} \tag{۲۶}$$

محاسبات را از بررسی یک عبارت ساده‌تر آغاز می‌کنیم:

$$\begin{aligned}\mathbb{E}(I_{u',v',w'}\sum_{u,v,w} I_{u,v,w}) &= \mathbb{P}(I_{u',v',w'} = ۱) \sum_{u,v,w} \mathbb{P}(I_{u,v,w} = ۱ | I_{u',v',w'} = ۱) \\ &= p^۳(۱ + p^۳ \times \binom{۳}{۱}\binom{n-۳}{۲} + p^۳ \times \binom{n-۳}{۳} + p^۲\binom{۳}{۲}\binom{n-۳}{۱}) \\ &(u' \neq v' \neq w')\end{aligned} \tag{۲۷}$$

حال عبارت خواسته شده را محاسبه می‌کنیم:

$$\begin{aligned}\mathbb{E}(T^۲_{۳,n}) &= \sum_{u',v',w'} I_{u',v',w'}(\sum_{u,v,w} I_{u,v,w}) \\ &= \sum_{\substack{u',v',w' \\ u'\neq v'\neq w'}} p^۳(۱ + ۳p^۳\binom{n-۳}{۲} + p^۳ \times \binom{n-۳}{۳} + ۳p^۲(n-۳)) \\ &= \binom{n}{۳}p^۳(۱ + ۳p^۳\binom{n-۳}{۲} + p^۳ \times \binom{n-۳}{۳} + ۳p^۲(n-۳)) \\ &\approx \frac{n^۳p^۳}{۶}(۱ + \frac{۳n^۲p^۳}{۲} + \frac{n^۳p^۳}{۶} + ۳np^۲) = \frac{n^۶p^۶}{۳۶} + \frac{n^۳p^۳}{۶} + \frac{n^۵p^۶}{۴} + \frac{n^۴p^۵}{۲}\end{aligned} \tag{۲۸}$$

۱۱.۵ پاسخ پرسش تئوری بیست و یکم

با فرض $p = c$ و با استفاده از ۲۸ می‌توان نوشت:

$$\begin{aligned}\frac{Var(T_{۳,n})}{\mathbb{E}(T_{۳,n})^۲} = \frac{\mathbb{E}(T^۲_{۳,n}) - \mathbb{E}(T_{۳,n})^۲}{\mathbb{E}(T_{۳,n})^۲} \Rightarrow \lim_{n\to\infty} \frac{Var(T_{۳,n})}{\mathbb{E}(T_{۳,n})^۲} &= \lim_{n\to\infty} \frac{\frac{n^۶c^۶}{۳۶} + \frac{n^۳c^۳}{۶} + \frac{n^۵c^۶}{۴} + \frac{n^۴c^۵}{۲} - (\frac{n^۳c^۳}{۶})^۲}{(\frac{n^۳c^۳}{۶})^۲} \\ &= \lim_{n\to\infty}(\frac{۶}{n^۳c^۳} + \frac{۹}{n} + \frac{۱۸c}{n^۲}) = ۰\end{aligned} \tag{۲۹}$$

۱۲.۵ پاسخ پرسش تئوری بیست و دوم

با توجه به رابطه ۲۳ و ۲۹ به راحتی می‌توان نتیجه گرفت:

$$\lim_{n\to\infty} P(T_{۳,n} \geq ۱) = ۱ \tag{۳۰}$$

که یعنی در گراف‌های با تعداد رئوس زیاد، حداقل یک مثلث درون گراف وجود دارد. شبیه‌سازی ۱۱ نیز همین نتیجه را به ما می‌دهد. در شبیهسازی، n از حدی که بیشتر شود، تعداد مثلث‌های میانگین از عدد ۱ بسیار بیشتر می‌شود، که یعنی اکثریت گراف‌ها تعداد مثلث نه فقط بیشتر از ۱، که تعداد زیادی مثلث دارند. (اگر میانگین تعداد مثلث‌ها کمتر یا نزدیک ۱ می‌شد به معنای آن بود که درصد قابل توجهی از گراف‌ها مثلث ندارند).

۱۳.۵ پاسخ پرسش تئوری بیست و سوم

اثبات این بخش، خود از اثبات دو قضیه تشکیل شده است.
قضیه اول: اگر متغیر تصادفی، توزیع پواسون داشته باشد، رابطه زیر برقرار است: $\mathbb{E}[(X)_r] = \lambda^r$ اثبات: ابتدا یک ویژگی کلی تابع مولد گشتاور فاکتوریل را اثبات می‌کنیم:

$$M_X(t) = \mathbb{E}(t^X) = \mathbb{E}(e^{X\ln t}) = \phi_X(s = \ln t) \tag{۳۱}$$

حال به اثبات اصلی می‌پردازیم:

$$P(X=k)=\frac{e^{-\lambda}\lambda^k}{k!}\Rightarrow \phi_{X(s)}=\sum_{k=0}^{\infty}\frac{e^{-\lambda}(\lambda e^s)^k}{k!}=e^{\lambda(e^s-1)}\Rightarrow \mathbb{E}(t^X)=e^{\lambda(e^{\ln t}-1)}=e^{\lambda t-\lambda}$$
$$\mathbb{E}[(X)_r]=\frac{\partial^r}{\partial t^r}e^{\lambda t-\lambda}\Big|_{t=1}=\lambda^r e^{\lambda t-\lambda}\Big|_{t=1}=\lambda^r \qquad (۳۲)$$

قضیه دوم: اگر رابطه $\mathbb{E}[(X)_r]=\lambda^r$ برقرار باشد، آنگاه متغیر تصادفی، توزیع پواسون دارد. اثبات: با فرض وجود ناحیه همگرایی، بسط تیلور تابع $M_X(t)$ را تشکیل می‌دهیم:

$$M_X(t)=\sum_{r=0}^{\infty}\frac{M^{(r)}(t=1)}{r!}(t-1)^r\Rightarrow M_X(t)=\sum_{r=0}^{\infty}\frac{(\lambda t-\lambda)^r}{r!}=e^{\lambda t-\lambda} \qquad (۳۳)$$

با استفاده از معادله ۳۱ داریم:

$$\phi_X(s)=e^{\lambda(e^s-1)} \qquad (۳۴)$$

حال با توجه به رابطه تابع مولد گشتاور بدست آمده در معادله ۳۲ و تناظر یک به یک توزیع متغیرهای تصادفی با تابع مولد گشتاور آن‌ها، می‌توان نتیجه گرفت که متغیر X توزیع پواسون دارد.

۱۴.۵ پاسخ پرسش تئوری بیست و چهارم

با توجه به رابطه ۲۱ و با جاگذاری $p=\frac{c}{n}$ داریم:

$$\mathbb{E}(T_{3,n})=\binom{n}{3}p^3=\frac{n(n-1)(n-2)}{6}\frac{c^3}{n^3}=\frac{c^3}{6}(1-\frac{1}{n})(1-\frac{2}{n})\xRightarrow{n\gg 1}\mathbb{E}(T_{3,n})\approx\frac{c^3}{6}=\lambda \qquad (۳۵)$$

همچنین با استفاده از معادله ۲۸ با جاگذاری $p=\frac{c}{n}$ به ازای $n\gg 1$ می‌توان نوشت:

$$\mathbb{E}[(X)_2]=\mathbb{E}[X(X-1)]=\mathbb{E}[X^2]-\mathbb{E}[X]\Rightarrow\mathbb{E}[(T_{3,n})_2]$$
$$\approx(\frac{n^6p^6}{36}+\frac{n^3p^3}{6}+\frac{n^5p^6}{4}+\frac{n^4p^5}{2})-\frac{n^3p^3}{6}=\frac{c^6}{36}+\frac{c^6}{4n}+\frac{c^5}{2n}\approx\frac{c^6}{36}=(\frac{c^3}{6})^2=\lambda^2 \qquad (۳۶)$$

۱۵.۵ پاسخ پرسش تئوری بیست و پنجم

۶ قومی به جد و جهد نهادند وصل دوست، قومی دگر حواله به تقدیر می کنند!

۱.۶ شبیه سازی سیزدهم

در این بخش، تابع اصلی کد از چند مرحله کلی تشکیل شده است. ابتدا به ازای هر گراف ساخته شده، مجموعه راس‌ها بدست آمده، سپس تعداد همسایگان (راس‌های مجاور) هر راس بدست می‌آید. بسته به صفر یا ناصفر بودن این مقادیر، تغییراتی در متغیرهای connectedp و isolation حاصل می‌شود. این روند به تعداد مشخصی که در ورودی گرفته می‌شود، تکرار می‌پذیرد.

```python
import numpy as np
import matplotlib.pyplot as mplot
from numpy.random import default_rng
import networkx as nx
```

```python

def func1(n,p,iteration):
    connected_p=0
    isolation=0
    for i in range(iteration):
        g=nx.fast_gnp_random_graph(n,p)
        nodes=g.nodes()
        p1=0
        for node in nodes:
            n1=g.neighbors(node)
            if (len(list(n1))==0):
                p1=p1+1/n
        if (p1==0):
            connected_p=connected_p+1
        isolation=isolation+p1
    return isolation/iteration,connected_p/iteration
print(func1(100,0.2,150))
```

برنامهٔ ۱۳: کد شبیه‌سازی دوازدهم

خروجی کد نیز به شرح زیر است:

```python
import numpy as np
import matplotlib.pyplot as mplot
from numpy.random import default_rng
import networkx as nx

def func1(n,p,iteration):
    connected_p=0
    isolation=0
    for i in range(iteration):
        g=nx.fast_gnp_random_graph(n,p)
        nodes=g.nodes()
        p1=0
        for node in nodes:
            n1=g.neighbors(node)
            if (len(list(n1))==0):
                p1=p1+1/n
        if (p1==0):
            connected_p=connected_p+1
        isolation=isolation+p1
    return isolation/iteration,connected_p/iteration
print(func1(100,0.2,150))
```

```
(0.0, 1.0)
```

شکل ۲۱: کد و خروجی شبیه‌سازی

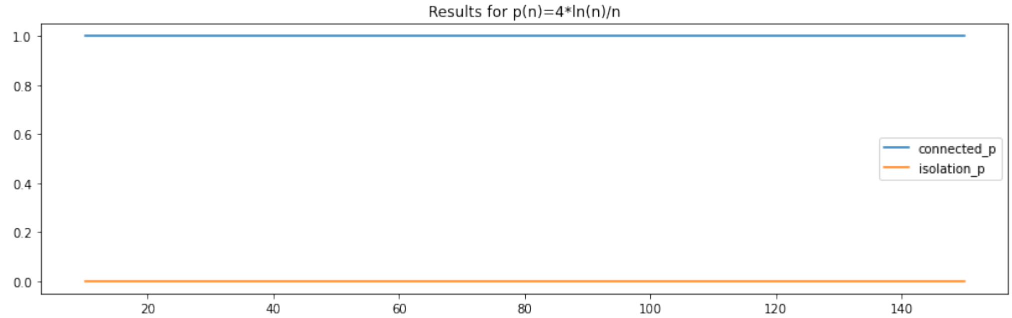
نتایج به ما نشان می‌دهند که احتمال صفر بودن درجه هر راس با تقریبب بسیار خوبی برابر صفر و احمال همبند بودن گراف تقریب ۱ می‌باشد. دقت داریم که همبند بودن گراف معادل حداثل درجه ۱ برای هر راس می‌باشد، که همخوانی بین خروجی‌ها را نشان می‌دهد.

۲.۶ شبیه سازی چهاردهم

در این بخش، کافیست از تابع بخش قبل استفاده کرده خروجی‌های متعدد از آن بگیریم. در خطوط ۲۱ تا ۲۷، ابتدا آرایه تعداد رئوس تشکیل شده سپس در یک حلقه، زوج خروجی تابع نوشته شده از هم جدا شده و هر کدام به یک آرایه جدا اضافه می‌شوند. خطوط بعدی نیز دستورات مربوط به نمودار هستند.

```
۱ import numpy as np
۲ import matplotlib.pyplot as mplot
۳ from numpy.random import default_rng
۴ import networkx as nx
۵
۶ def isolation_connection(n,p,iteration):
۷     connected_p=0
۸     isolation=0
۹     for i in range(iteration):
۱۰         g=nx.fast_gnp_random_graph(n,p)
۱۱         nodes=g.nodes()
۱۲         p1=0
۱۳         for node in nodes:
۱۴             n1=g.neighbors(node)
۱۵             if (len(list(n1))==0):
۱۶                 p1=p1+1/n
۱۷         if (p1==0):
۱۸             connected_p=connected_p+1
۱۹         isolation=isolation+p1
۲۰     return isolation/iteration,connected_p/iteration
۲۱ x=np.arange(10,160,10)
۲۲ connected_p=np.array([])
۲۳ isolation_p=np.array([])
۲۴ for n in x:
۲۵     a,b=isolation_connection(n,4*np.log(n)/n,150)
۲۶     connected_p=np.append(connected_p,[b])
۲۷     isolation_p=np.append(isolation_p,[a])
۲۸ fig1,ax1=mplot.subplots(figsize=(14, 4))
۲۹ ax1.set_title('Results for p(n)=4*ln(n)/n')
۳۰ mplot.plot(x,connected_p,label="connected_p")
۳۱ mplot.plot(x,isolation_p,label="isolation_p")
۳۲ mplot.legend()
```

برنامهٔ ۱۴: کد شبیه‌سازی دوازدهم



شکل ۲۲: نمودار خروجی شبیه‌سازی

مشاهدات ما نشان می‌دهند که احتمال همبند بودن ۱ و احتمال صفر بودن درجات صفر می‌باشد. طبیعتا وقتی گراف همبند باشد تمام رئوس درجه حداقل ۱ را دارا هستند. در نتیجه، نتایج شبیه‌سازی با هم سازگارند. همچنین، در بخش ۲۹ تئوری نشان داده می‌شود که تئوری احتمال نیز همین نتیجه را به ما می‌دهد.

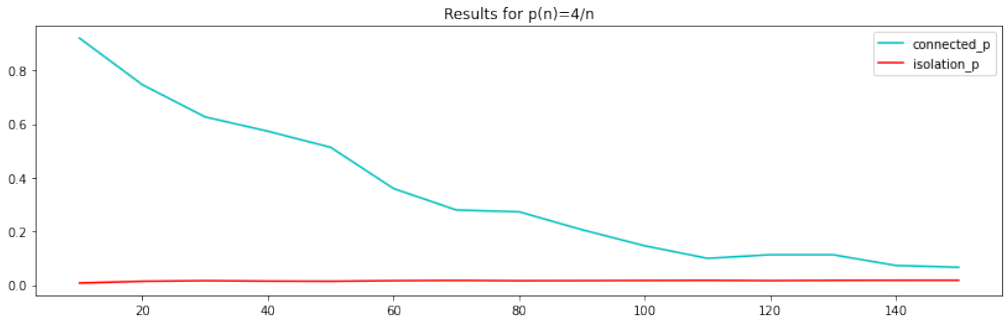
۳.۶ شبیه سازی پانزدهم

کد این بخش با شبیه سازی چهاردهم هیچ تفاوتی بجز رابطه $P(n) = \frac{۴}{n}$ ندارد.

```
import numpy as np
import matplotlib.pyplot as mplot
from numpy.random import default_rng
import networkx as nx

def isolation_connection(n,p,iteration):
    connected_p=0
    isolation=0
    for i in range(iteration):
        g=nx.fast_gnp_random_graph(n,p)
        nodes=g.nodes()
        p1=0
        for node in nodes:
            n1=g.neighbors(node)
            if (len(list(n1))==0):
                p1=p1+1/n
        if (p1==0):
            connected_p=connected_p+1
        isolation=isolation+p1
    return isolation/iteration,connected_p/iteration
x=np.arange(10,160,10)
connected_p1=np.array([])
isolation_p1=np.array([])
for n in x:
    a,b=isolation_connection(n,4/n,150)
    connected_p1=np.append(connected_p1,[b])
    isolation_p1=np.append(isolation_p1,[a])
fig2,ax2=mplot.subplots(figsize=(14, 4))
```

```python
ax2.set_title('Results for p(n)=4/n')
mplot.plot(x,connected_p1,'c',label="connected_p")
mplot.plot(x,isolation_p1,'r',label="isolation_p")
mplot.legend()
```

برنامهٔ ۱۵: کد شبیه‌سازی دوازدهم



شکل ۲۳: نمودار خروجی شبیه‌سازی

نتایج این نمودار به ما نشان می‌دهند که احتمال همبند بودن هموار بسیار کم است ولی با تعداد رئوس زیاد، احتمال منفرد بودن راس‌ها نیز کاهش می‌یابد که به معنای زیاد بودن تشکیل زیرگراف‌های پیوسته می‌باشد.

۴.۶ پاسخ پرسش تئوری بیست و ششم

دو پیشامد $A_j^{(i)}$ و $B_j^{(i)}$ را به شرح زیر تعریف می‌کنیم:

$A_i^{(j)} =$ عدم اتصال هیچ یک از راس j به سایر راس‌ها

$B_i^{(j)} =$ همبند بودن زیرگراف حاصل از انتخاب j راس

$C_{i,\ell}^{(j)} =$ عدم اتصال راس شماره ℓ به راس‌های خارج از زیرگراف مشخص شده $\forall \ell \in [۱, j]$

با توجه به استقلال تشکیل شدن یال‌ها از همدیگر، می‌توان نوشت:

$$\begin{aligned}
&\mathbb{P}(K_i^{(j)} = ۱) = \mathbb{P}(A_j^{(i)})\mathbb{P}(B_j^{(i)}) \leq \mathbb{P}(A_j^{(i)}) \times ۱ = \mathbb{P}(A_j^{(i)}) \\
&\mathbb{P}(C_{i,\ell}^{(j)}) = (۱ - p)^{n-j},\ \mathbb{P}(A_i^{(j)}) = \mathbb{P}(C_{i,\ell}^{(j)})^j \\
&\Rightarrow \mathbb{P}(A_i^{(j)}) = (۱ - p)^{j(n-j)} \Rightarrow \mathbb{P}(K_i^{(j)} = ۱) \leq (۱ - p)^{j(n-j)}
\end{aligned} \tag{۳۷}$$

۵.۶ پاسخ پرسش تئوری بیست و هفتم

با استفاده از نابرابری بدست آمده در رابطه ۳۷ و خواص امیدریاضی می‌توان نوشت:

$$\begin{aligned}
X_j = \sum_{i=0}^{\binom{n}{j}} K_i^{(j)} &\Rightarrow \mathbb{E}[X_j] = \sum_{i=0}^{\binom{n}{j}} \mathbb{E}[K_i^{(j)}] = \sum_{j=0}^{\binom{n}{j}} \mathbb{P}(K_i^{(j)} = 1) \\
&\leq \sum_{i=0}^{\binom{n}{j}} (1-p)^{j(n-j)} = \binom{n}{j}(1-p)^{j(n-j)} \\
\Rightarrow \mathbb{E}[X_j] &\leq \binom{n}{j}(1-p)^{j(n-j)}
\end{aligned} \tag{38}$$

۶.۶ پاسخ پرسش تئوری بیست و هشتم

با استفاده مجدد از خاصیت خطی بودن امیدریاضی و با استفاده از نامساوی قسمت قبل، می‌توان به نامساوی دیگری رسید. می‌دانیم به ازای هر $n \gg 1$ و $p = \frac{a \ln n}{n}$ با شرط a ۳ چنین رابطه‌ای برقرار است:

$$\binom{n}{j}(1-p)^{j(n-j)} \leq \frac{1}{n^{1/7}} \tag{39}$$

با استفاده از این رابطه، داریم:

$$X = \sum_{j=0}^{n} X_j \Rightarrow \mathbb{E}[X] = \sum_{j=0}^{n} \mathbb{E}[X_j] \leq \sum_{j=0}^{n} \binom{n}{j}(1-p)^{j(n-j)} \leq \sum_{j=0}^{n} \frac{1}{n^{1/7}} = \frac{n}{n^{1/7}} = \frac{1}{n^{0/7}} \tag{40}$$

بررسی کران در حد میل کردن n به بینهایت:

$$\lim_{n\to\infty} \frac{1}{n^{0/7}} = 0$$

۷.۶ پاسخ پرسش تئوری بیست و نهم

همچنین با توجه به تعریف متغیر تصادفی X که از جنس تعداد است، می‌دانیم $X \geq 0$ و در نتیجه $\mathbb{E}[X] \geq 0$. با توجه به این موضوع و استفاده از قضیه فشار در حد، به روابط زیر می‌رسیم:

$$0 \leq \mathbb{E}[X] \leq \frac{1}{n^{0/7}}, \lim_{n\to\infty} \frac{1}{n^{0/7}} = 0 \Rightarrow \lim_{n\to\infty} \mathbb{E}[X] = 0 \tag{41}$$

با استفاده از نامساوی مارکف اثبات را تکمیل می‌کنیم:

$$\begin{aligned}
0 \leq \mathbb{P}(X \geq 2) \leq \frac{E(X)}{2} \Rightarrow, \lim_{n\to\infty} \mathbb{E}[X] = 0 \Rightarrow \lim_{n\to\infty} \mathbb{P}(X \geq 2) = 0 \\
\mathbb{P}(\text{هم‌بند بودن گراف}) = \mathbb{P}(X = 0) = 1 - \mathbb{P}(X \geq 2) \Rightarrow \lim_{n\to\infty} \mathbb{P}(\text{هم‌بند بودن گراف}) = \mathbb{P}(X = 0) = 1
\end{aligned} \tag{42}$$