

در ابتدای کار با کتاب‌خانه‌های مختلفی کار کردم. اما در نهایت از کتاب‌خانه‌ی `networkx` در پایتون استفاده کردم.

در این کتابخانه در `networkx.utils` می‌توان یک لیست رندوم از توزیع `powerlaw` تولید کرد. به صورتی که با هر بار فراخوانی `powerlaw_sequence` یک لیست رندوم به شما خروجی می‌دهد که در اصل نودها و تعداد یال‌ها و... گراف را برای ما تداعی می‌کنند. ورودی تابع، تعداد نودهایی که ما می‌خواهیم + لاندا یا همان `exponent` ما است که قابل تنظیم است.

در مرحله‌ی بعد باید با توجه به اطلاعات تولید شده در مرحله‌ی قبل گراف خود را تشکیل دهیم. پس خروجی مرحله‌ی قبل را باید تبدیل به یک گراف کنیم تا بتوانیم اطلاعات مورد نیاز را از آن استخراج کنیم. خوشبختانه در همان کتاب‌خانه `networkx` در بخش `generators` تابعی به نام `expected_degree_graph` وجود دارد که گراف ما را با توجه به خروجی قسمت قبل برای ما می‌سازد.

در مرحله‌ی بعد، به دلیل اینکه ما به `giant_component` در گرافمان نیاز داریم و نمی‌خواهیم نودهایی مانند نودهای ایزوله در گرافمان موجود باشند، با استفاده از `remove_nodes_from` که یک تابع از کلاس `Graph` از این کتابخانه است، نودهای ایزوله را از گراف اصلی خود حذف می‌کنیم. (با دادن لیست نودهای ایزوله به این تابع)

در مرحله‌ی بعد ما شاهد چندین `subgraph` خواهیم بود که `giant component` در اصل بزرگ‌ترین `subgraph` ما است. حال این `subgraph`‌ها را بر حسب سائزشان مرتب می‌کنیم و بزرگ‌ترین آن‌ها را انتخاب می‌کنیم و سپس برای بزرگ‌ترین آن‌ها به دنبال میانگین طول کوتاه‌ترین مسیر در گراف می‌گردیم.

خوشبختانه کتابخانه‌ی `networkx` این مرحله را نیز پوشش داده است و از تابع `average_shortest_path_length` می‌توانیم استفاده کنیم که به عنوان ورودی یک گراف از شما می‌گیرد (که همان `giant component` را به عنوان ورودی به آن می‌دهیم) و در نهایت یک عدد به شما برمی‌گرداند که همان $\langle k \rangle$ که تمرین از ما خواسته است، می‌باشد.

برای مثال: همانطور که در شکل زیر می بینید قطعه کد را برای ۱۰۰ نود با لاندا ۲-۲.۵-۳ و ۵ اجرا کردیم و برای هر کدام ۳ بار $\langle d \rangle$ را محاسبه کردیم. میانگین این لیست به طور تقریبی به جواب اصلی مسئله نزدیک است.(به علت کمبود منابع برای نودهایی با درجات بسیار بالاتر این امکان وجود نداشت.)

```
for node and lambda: (100, 2) <d> in 10 times try: [2.428786510087323, 2.738282375851996, 2.278283861766213, 2.73792312105565, 2.187889533952954, 2.565811965811966, 2.3438974789915967, 2.82731889869019, 2.812960143334926, 2.6256244496155747] a
verage <d> is: 2.47143675466045
for node and lambda: (100, 2.5) <d> in 10 times try: [3.170189222829802, 3.329473684210526, 3.6641137525712666, 3.275910364145658, 2.8094298245614035, 2.369883447332421, 3.4245754245754245, 3.620984278879016, 2.9782544813399943, 3.1094736842105264]
average <d> is: 3.0951797164656036
for node and lambda: (100, 3) <d> in 10 times try: [4.271689497716895, 3.6572769953851645, 5.28969857218403, 2.5856962025316456, 4.539344262295082, 3.777403846153846, 2.169237961664329, 3.810175438596491, 3.681999259533506, 4.7518796992481205] av
erage <d> is: 3.835440173522911
for node and lambda: (100, 5) <d> in 10 times try: [4.82740846153846, 5.385365853658537, 4.485294117647059, 5.490950226244344, 6.260109289617486, 7.781970649895178, 7.117532467532468, 7.7481481481481485, 8.579787234842554, 9.463934426229509] ave
rage <d> is: 6.714049625916914
for node and lambda: (500, 2) <d> in 10 times try: [2.1923641251372343, 2.9404388801578833, 2.731793787607741, 2.1218988822336673, 2.382624634334103, 2.033887668327289, 2.9219911353562904, 2.7326692394885994, 2.83728936788983, 2.8937641535487435]
average <d> is: 2.5788120256973387
for node and lambda: (500, 2.5) <d> in 10 times try: [3.5139135504739685, 3.729221061792853, 3.805727298170623, 4.19277303007873, 5.003061493766665, 3.853548878461047, 4.084919360020562, 4.713338006958509, 4.5468157926903805, 3.4151387290627797]
average <d> is: 4.085839720147522
for node and lambda: (500, 3) <d> in 10 times try: [5.409843971631206, 5.618682069947776, 6.061933932747094, 3.26126862958924, 6.094128251352847, 5.618229966072025, 6.3765737559089257, 5.355915065722953, 4.629551451187335, 4.698827129015808] avera
ge <d> is: 5.311695322235474
for node and lambda: (500, 5) <d> in 10 times try: [10.951845431869108, 10.340922904296601, 10.740844621513943, 7.5838986883763, 11.133798004646714, 11.354194115063681, 10.598506568516528, 12.669673306772909, 9.990411731528482, 12.827775432672013]
average <d> is: 10.818392480525628
for node and lambda: (1000, 2) <d> in 10 times try: [2.449453238406445, 2.87249098186101, 2.262925563229794, 2.973511751586893, 2.996198961406356, 2.977109938134968, 2.979990720174834, 2.5301368871265955, 2.6758486032641557, 2.6492182812187477]
average <d> is: 2.7206804042713007
for node and lambda: (1000, 2.5) <d> in 10 times try: [4.440653641986855, 2.2342291083524997, 4.405284571918841, 4.469001521296501, 5.2121443304795, 3.1930405965202984, 3.542473484503722, 3.595154141633015, 3.7267258515952184, 4.253850149465261]
average <d> is: 3.9072557309661717
for node and lambda: (1000, 3) <d> in 10 times try: [5.480808555550059, 6.3990361781763285, 6.265559018123918, 6.831095187022916, 6.206868551239422, 6.204485587715291, 6.07868857139393, 6.195841187103644, 4.962913676702928, 6.0689698378619425] av
erage <d> is: 5.983425030208947
for node and lambda: (1000, 5) <d> in 10 times try: [11.179298687803696, 10.915150095900799, 12.515575190575191, 12.094109943513523, 11.437877964874163, 13.624861145949046, 12.727861598855651, 11.773638143498792, 12.483473053892215, 13.68196361819
337] average <d> is: 12.243380942625844
for node and lambda: (2000, 2) <d> in 10 times try: [2.2515774754217577, 2.545700963100472, 2.0692312216928417, 3.024981831140443, 3.1588886118334343, 2.7382318158584598, 2.999898237375074, 3.0397785904691688, 2.2541502532357907, 3.114379579503351]
average <d> is: 2.7196818579630793
for node and lambda: (2000, 2.5) <d> in 10 times try: [5.103595147724896, 4.9235997310015645, 3.7661005684186426, 3.98637505389332, 4.1735742597575705, 3.8237135052115607, 3.587665832320123, 4.454044440742206, 5.082320468633025, 4.98952817168627]
average <d> is: 4.389084120911558
for node and lambda: (2000, 3) <d> in 10 times try: [6.3412328126158095, 5.748973647963402, 5.600865424926633, 6.916895918611543, 6.14468045629611, 6.016778048607136, 6.263435681974731, 5.185701253090183, 5.5755106902864995, 6.141609932481133] a
verage <d> is: 5.993597064018667
for node and lambda: (2000, 5) <d> in 10 times try: [12.273071589426507, 14.655958537261059, 13.664417788800634, 13.214681368343324, 16.100421742871976, 16.966651946510634, 13.458669367665884, 14.632246568673901, 14.631881881881881, 15.84261311717
8672] average <d> is: 14.544061390861248
for node and lambda: (4000, 2) <d> in 10 times try: [2.5920972188549584, 2.9566090942728533, 2.0236373620914434, 2.832930248770126, 2.7038621633391826, 2.115516184786026, 3.1373653452086525, 2.4508707431123865, 2.5582015370974993, 2.5881181026499
82] average <d> is: 2.5958400906990047
for node and lambda: (4000, 2.5) <d> in 10 times try: [3.9585068688757117, 4.272836191948526, 4.390921763010891, 4.1205337824722915, 4.894447592704342, 4.899387838756047, 3.9039827937393006, 4.647294476933627, 3.9149129892766914, 2.33411567553383]
average <d> is: 4.133638740040859
for node and lambda: (4000, 3) <d> in 10 times try: [6.256954833445104, 5.964622594389615, 6.912486209871155, 5.89695000945652, 6.547986315161981, 6.6595192307206075, 6.63614500105621, 6.282630761674662, 6.4923293092493415, 4.890220544656385] av
erage <d> is: 6.254158980117063
for node and lambda: (4000, 5) <d> in 10 times try: [14.718735296117496, 16.998048217371867, 16.063341189274606, 17.42480065355717, 17.137413025651433, 16.531235173967108, 16.582298158827165, 15.708037114664746, 15.288578665727273, 18.203476884107
868] average <d> is: 16.45759643792667
for node and lambda: (6000, 2) <d> in 10 times try: [2.5263981661925348, 3.064941081844132, 2.6154650592998854, 2.601110130354957, 3.118184116450861, 3.022145249847936, 2.313997704838028, 2.306392336908915, 2.442043468313619, 2.7945718285902013]
average <d> is: 2.6797249142641073
for node and lambda: (6000, 2.5) <d> in 10 times try: [3.9467479813987687, 4.129364315718334, 4.336680551931967, 3.1907295575975207, 3.615913115618386, 4.163916541453963, 4.976589213558221, 4.960866457563348, 4.632532995722605, 4.327583887623239]
average <d> is: 4.228076453758636
for node and lambda: (6000, 3) <d> in 10 times try: [5.766996978244039, 5.586376320797202, 6.805844294623649, 6.409883266364695, 7.676896682935627, 6.736827434842249, 7.268088199939976, 7.338834252818422, 6.553106955816311, 6.06674783157609] ave
rage <d> is: 6.620960221794879
for node and lambda: (6000, 5) <d> in 10 times try: [17.14936911953361, 15.79665156031812, 16.95769932214903, 19.19636976201488, 17.42023223821808, 17.169040590484284, 17.50997025434268, 18.688595139866532, 17.1235261030909826, 17.470336357152508]
average <d> is: 17.447279044708946
[(100, 2): 2.47143675466045, (100, 2.5): 3.0951797164656036, (100, 3): 3.835440173522911, (100, 5): 6.714049625916914, (500, 2): 2.5788120266973387, (500, 2.5): 4.085839720147522, (500, 3): 5.311695322235474, (500, 5): 10.818392480525628, (1000, 2
): 2.7306884942733087, (1000, 2.5): 3.9072557309661717, (1000, 3): 5.983425030208947, (1000, 5): 12.243380942625844, (2000, 2): 2.7196818579630793, (2000, 2.5): 4.389084120911558, (2000, 3): 5.993597064018667, (2000, 5): 14.544061390861248, (4000,
2): 2.5958400906990047, (4000, 2.5): 4.133638740040859, (4000, 3): 6.254158980117063, (4000, 5): 16.45759643792667, (6000, 2): 2.6797249142641073, (6000, 2.5): 4.228076453758636, (6000, 3): 6.620960221794879, (6000, 5): 17.447279044708946]
```

حال در مرحله‌ی بعد اطلاعات خروجی را به نمودار می‌بریم.
برای مثال برای لاندای ۲.۵ و لاندای ۵ دو نمودار زیر را داریم. که با توجه به اضافه شدن درجه‌ی
رئوس، $\langle d \rangle$ نیز سیری افزایشی خواهد داشت.



