Large Scale Social and Complex Networks: Design and Algorithms
ECE 232E Spring 2023
Prof. Vwani Roychowdhury
UCLA, Department of ECE

Project 1
Random Graphs and Random Walks

Due on Saturday, April 22, 2023 by 11:59 PM PST uploaded via BruinLearn

One can use `igraph` library[1] to generate different networks and measure various properties of a given network. The library has `R` and `Python` implementations. You may choose either language that you prefer. However, for this project, using `R` is recommended, as some functionality might not be implemented for the `Python` version of the package.

Submission: Please submit your report to Gradescope for grading. Meanwhile, upload a zip file containing your report and codes to BruinLearn Assignments. **Name the submission with your group members' UIDs. One submission per group only**.

[1] http://igraph.org/

1. Generating Random Networks

1. Create random networks using Erdös-Rényi (ER) model

   (a) Create undirected random networks with $n = 900$ nodes, and the probability p for drawing an edge between two arbitrary vertices 0.002, 0.006, 0.012, 0.045, and 0.1. Plot the degree distributions. What distribution (linear/exponential/gaussian/binomial or something else) is observed? Explain why. Also, report the mean and variance of the degree distributions and compare them to the theoretical values.
   Hint Useful function(s): `samplegnp (erdos.renyi.game)`, `degree`, `degreedistribution`, `plot`

   (b) For each p and $n = 900$, answer the following questions:
   Are all random realizations of the ER network connected? Numerically estimate the probability that a generated network is connected. For one instance of the networks with that p, find the giant connected component (GCC) if not connected. What is the diameter of the GCC?
   Hint Useful function(s): `isconnected`, `clusters`, `diameter`

   (c) It turns out that the normalized GCC size (i.e., the size of the GCC as a fraction of the total network size) is a highly nonlinear function of p, with interesting properties occurring for values where $p = O(\frac{1}{n})$ and $p = O(\frac{\ln n}{n})$.
   For $n = 900$, sweep over values of p from 0 to a $p_{\max}$ that makes the network almost surely connected and create 100 random networks for each p. $p_{\max}$ should be roughly determined by yourself. Then scatter plot the normalized GCC sizes vs p. Plot a line of the average normalized GCC sizes for each p along with the scatter plot.

      i. Empirically estimate the value of p where a giant connected component starts to emerge (define your criterion of "emergence")? Do they match with theoretical values mentioned or derived in lectures?

      ii. Empirically estimate the value of p where the giant connected component takes up over 99% of the nodes in almost every experiment.

   (d) i. Define the average degree of nodes $c = n \times p = 0.5$. Sweep over the number of nodes, n, ranging from 100 to 10000. Plot the expected size of the GCC of ER networks with n nodes and edge-formation probabilities $p = c/n$, as a function of n. What trend is observed?

      ii. Repeat the same for $c = 1$.

      iii. Repeat the same for values of $c = 1.15, 1.25, 1.35$, and show the results for these three values in a single plot.

      iv. What is the relation between the expected GCC size and n in each case?

2. Create networks using preferential attachment model

   (a) Create an undirected network with $n = 1050$ nodes, with preferential attachment model, where each new node attaches to $m = 1$ old nodes. Is such a network always connected?
   Hint Useful function(s): `samplepa (barabasi.game)`

   (b) Use fast greedy method to find the community structure. Measure modularity. Define Assortativity. Compute Assortativity.
   Hint Useful function(s): `clusterfastgreedy`, `modularity`

   (c) Try to generate a larger network with 10500 nodes using the same model. Compute modularity and assortativity. How is it compared to the smaller network's modularity?

(d) Plot the degree distribution in a log-log scale for both $n = 1050, 10500$, then estimate the slope of the plot using linear regression.

(e) In the two networks generated in 2(a) and 2(c), perform the following:
Randomly pick a node i, and then randomly pick a neighbor j of that node. Plot the degree distribution of nodes j that are picked with this process, in the log-log scale. Is the distribution linear in the log-log scale? If so, what is the slope? How does this differ from the node degree distribution?
Hint Useful function(s): `sample`

(f) Estimate the expected degree of a node that is added at time step i for $1 \leq i \leq 1050$. Show the relationship between the age of nodes and their expected degree through an appropriate plot. Note that the newest added node is the youngest.

(g) Repeat the previous parts (a-f) for $m = 2$, and $m = 6$. Compare the results of each part for different values of m.

(h) Again, generate a preferential attachment network with $n = 1050$, $m = 1$. Take its degree sequence and create a new network with the same degree sequence, through stub-matching procedure. Plot both networks, mark communities on their plots, and measure their modularity. Compare the two procedures for creating random power-law networks.
Hint In case that fastgreedy community detection fails because of self-loops, you may use "walktrap" community detection.
Useful function(s): `sampledegseq`

3. Create a modified preferential attachment model that penalizes the age of a node

(a) Each time a new vertex is added, it creates m links to old vertices and the probability that an old vertex is cited depends on its degree (preferential attachment) and age. In particular, the probability that a newly added vertex connects to an old vertex is proportional to:

$$P[i] \sim (ck_i^{\alpha} + a)(dl_i^{\beta} + b),$$

where k_i is the degree of vertex i in the current time step, and l_i is the age of vertex i. Produce such an undirected network with 1050 nodes and parameters $m = 1$, $\alpha = 1, \beta = -1$, and $a = c = d = 1, b = 0$. Plot the degree distribution. What is the power law exponent?
Hint Useful function(s): `samplepaage`

(b) Use fast greedy method to find the community structure. What is the modularity?

2. Random Walk on Networks

1. Random walk on Erdös-Rényi networks

   (a) Create an undirected random network with 900 nodes, and the probability p for drawing an edge between any pair of nodes equal to 0.015.

   (b) Let a random walker start from a randomly selected node (no teleportation). We use t to denote the number of steps that the walker has taken. Measure the average distance (defined as the shortest path length) $\langle s(t) \rangle$ of the walker from his starting point at step t. Also, measure the variance $\sigma^2(t) = \langle (s(t) - \langle s(t) \rangle)^2 \rangle$ of this distance. Plot $\langle s(t) \rangle$ v.s. t and $\sigma^2(t)$ v.s. t. Here, the average $\langle \cdot \rangle$ is over random choices of the starting nodes.

   (c) Measure the degree distribution of the nodes reached at the end of the random walk. How does it compare to the degree distribution of graph?

   (d) Repeat 1(b) for undirected random networks with 9000 nodes. Compare the results and explain qualitatively. Does the diameter of the network play a role?

2. Random walk on networks with fat-tailed degree distribution

   (a) Generate an undirected preferential attachment network with 900 nodes, where each new node attaches to $m = 1$ old nodes.

   (b) Let a random walker start from a randomly selected node. Measure and plot $\langle s(t) \rangle$ v.s. t and $\sigma^2(t)$ v.s. t.

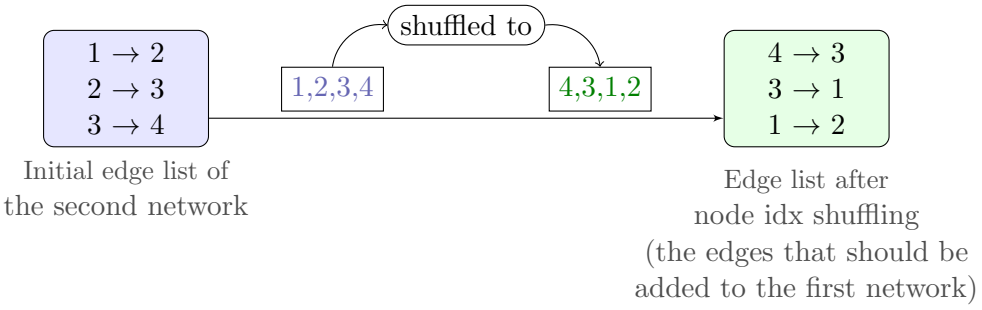
   (c) Measure the degree distribution of the nodes reached at the end of the random walk on this network. How does it compare with the degree distribution of the graph?

   (d) Repeat 2(b) for preferential attachment networks with 90 and 9000 nodes, and $m = 1$. Compare the results and explain qualitatively. Does the diameter of the network play a role?

3. PageRank

   The PageRank algorithm, as used by the Google search engine, exploits the linkage structure of the web to compute global "importance" scores that can be used to influence the ranking of search results. Here, we use random walk to simulate PageRank.

   (a) We are going to create a directed random network with 900 nodes, using the preferential attachment model. Note that in a directed preferential attachment network, the out-degree of every node is m, while the in-degrees follow a power law distribution. One problem of performing random walk in such a network is that, the very first node will have no outbounding edges, and be a "black hole" which a random walker can never "escape" from. To address that, let's generate another 900-node random network with preferential attachment model, and merge the two networks by adding the edges of the second graph to the first graph with a shuffling of the indices of the nodes. For example,



Create such a network using $m = 4$. Measure the probability that the walker visits each node. Is this probability related to the degree of the nodes?

Hint Useful function(s): `asedgelist` , `sample` , `permute` , `addedges`

(b) In all previous questions, we didn't have any teleportation. Now, we use a teleportation probability of $\alpha = 0.2$ (teleport out of a node with prob=0.2 instead of going to its neighbor). By performing random walks on the network created in 3(a), measure the probability that the walker visits each node. How is this probability related to the degree of the node and α ?

4. Personalized PageRank

While the use of PageRank has proven very effective, the web's rapid growth in size and diversity drives an increasing demand for greater flexibility in ranking. Ideally, each user should be able to define their own notion of importance for each individual query.

(a) Suppose you have your own notion of importance. Your interest in a node is proportional to the node's PageRank, because you totally rely upon Google to decide which website to visit (assume that these nodes represent websites). Again, use random walk on network generated in question 3 to simulate this personalized PageRank. Here the teleportation probability to each node is proportional to its PageRank (as opposed to the regular PageRank, where at teleportation, the chance of visiting all nodes are the same and equal to $\frac{1}{N}$). Again, let the teleportation probability be equal to $\alpha = 0.2$. Compare the results with 3(a).

(b) Find two nodes in the network with median PageRanks. Repeat part 4(a) if teleportations land only on those two nodes (with probabilities 1/2, 1/2). How are the PageRank values affected?

(c) More or less, 4(b) is what happens in the real world, in that a user browsing the web only teleports to a set of trusted web pages. However, this is against the assumption of normal PageRank, where we assume that people's interest in all nodes are the same. Can you take into account the effect of this self-reinforcement and adjust the PageRank equation?

Final Remarks

The following functions from igraph library are useful for this project:

- `degree, degree.distribution, diameter, clusters, vcount, ecount`
- `random.graph.game, barabasi.game, aging.prefatt.game, degree.sequence.game`
- `page_rank`

For part 2 of the project, you can start off with the Jupyter notebook provided to you.