Network Models

Michael Goodrich

Some slides adapted from:
Networked Life (NETS) 112, Univ. of Penn., 2018, Prof. Michael Kearns
Efficient generation of large random networks, by Batagelj and Brandes.
Database of Real-World Graphs

- **SNAP**: Stanford Network Analysis Project’s Large Network Dataset Collection
- [http://snap.stanford.edu/data/index.html](http://snap.stanford.edu/data/index.html)
- Many real-world networks:
  - Social networks: online social networks, edges represent interactions between people
  - Communication networks: email communication networks with edges representing communication
  - Citation networks: nodes represent papers, edges represent citations
  - Collaboration networks: nodes represent scientists, edges represent collaborations (co-authoring a paper)
  - Web graphs: nodes represent webpages and edges are hyperlinks
  - Amazon networks: nodes represent products and edges link commonly co-purchased products
  - Internet networks: nodes represent computers and edges communication
  - ... many more
Network Models

• Recent studies of complex systems such as the Internet, biological networks, or social networks, have significantly increased the interest in modeling networks.

• Network models are desired that match real-world graph structures and properties, including:
  – Degree distributions
  – Small-world property
  – Clustering coefficients

Image from https://matrix.berkeley.edu/research/social-networks-history
Network Models

I. The Erdös-Rényi (Random Graph) Model
Random Graphs (Erdös/Rényi)

- **$G(n,p)$:**
  - $n$ nodes
  - Every pair of nodes is connected independently with probability $p$
  - Average degree: $d = (n-1)p \sim np$
Erdös-Rényi G(n,p) Generation

• Begin with n isolated vertices, no edges
• Consider (unordered) vertex pairs, \{v,w\}, according to some ordering.
• For each such pair, \{v,w\}:
  – Randomly generate a bit, b, that is 1 with probability p.
  – If b = 1, then add the edge (v,w) to the graph

• This algorithm runs in $O(n^2)$ time, however.
Faster Erdös-Rényi G(n,p) Generation

- The above algorithm for generating G(n,p) is slow if p is small, because most of the bits are 0.
- Probability of having k-1 0’s then a 1 is $q^{k-1}p$, where $q = 1-p$.
- Waiting times are geometrically distributed.
- Divide the interval [0,1) according to the waiting times:

```
(0, p)         (qp, q^2p)          (1)
```

Faster Erdös-Rényi $G(n,p)$ Generation

- Pick $r$ uniformly at random in the interval $[0,1)$
- Divide the interval $[0,1)$ according to the waiting times.
- The subinterval in which $r$ falls will sample a waiting time:

Note that

$r < 1 - q^k \iff k > \frac{\log(1-r)}{\log q}$,

so that we choose $k = 1 + [\log(1-r)/\log q]$. 
Faster Erdös-Rényi $G(n,p)$ Generation

• The above algorithm for generating $G(n,p)$ is slow if $p$ is small, because most of the bits are 0.
• Probability of having $k-1$ 0’s then a 1 is $(1-p)^{k-1}p$
• Faster $O(n+m)$-time algorithm skips over runs of 0’s:

```
ALG. 1: $G(n,p)$
Input: number of vertices $n$, edge probability $0 < p < 1$
Output: $G=(\{0, \ldots, n-1\},E) \in G(n,p)$

$E \leftarrow \emptyset$

$v \leftarrow 1$; $w \leftarrow -1$

while $v < n$ do
  draw $r \in [0,1)$ uniformly at random
  $w \leftarrow w + 1 + [\log(1-r)/\log(1-p)]$
  while $w \geq v$ and $v < n$ do
    $w \leftarrow w - v$; $v \leftarrow v + 1$
  if $v < n$ then $E \leftarrow E \cup \{v,w\}$
```
There Can’t Be Two Large Components?

\[ \frac{N}{2} \]

densely connected

\[ \frac{N^2}{4} \]

missing edges

\[ \frac{N}{2} \]

densely connected
Threshold Phenomena in Erdös-Rényi

• Theorem: In Erdös-Rényi, as \( n \) becomes large:
  - If \( p < \frac{1}{n} \), probability of a giant component (e.g. 50% of vertices) goes to 0
  - If \( p > \frac{1}{n} \), probability of a giant component goes to 1, and all other components will have size at most \( \log(n) \)

• Note: at edge density \( p \), expected/average degree is \( p(N-1) \sim pn \)

• So at \( p \sim \frac{1}{n} \), average degree is \( \sim 1 \): incredibly sparse
• So model “explains” giant components in real networks

• General “tipping point” at edge density \( q \) (may depend on \( n \)):
  - If \( p < q \), probability of property goes to 0 as \( n \) becomes large
  - If \( p > q \), probability of property goes to 1 as \( n \) becomes large

• For example, could examine property “diameter 6 or less”
Threshold Phenomena in Erdős-Renyi

• Theorem: In Erdős-Renyi, as N becomes large:
  – The diameter is \( O(\log(N) / \log(Np) \).
  – Threshold at
    \[
    p \sim \frac{\log(N)}{N^{5/6}}
    \]
    – for diameter 6.
  – Note: degrees growing (slightly) with N
    – If N = 300M (U.S. population) then average degree \( pN \sim 500 \)
    – If N = 7BN (world population) then average degree \( pN \sim 1000 \)
    – Not unreasonable figures…

• At \( p \) not too far from \( 1/N \), get strong connectivity
• Very efficient use of edges
What Doesn’t the Model Explain?

- Erdös-Renyi explains giant component and small diameter
- But:
  - degree distribution not heavy-tailed; exponential decay from mean (Poisson)
  - clustering coefficient is *exactly* $p$

- To model these real-world phenomena, we’ll need richer models with greater realism…
Rich-Get-Richer Processes

- Processes in which the more someone has of something, the more likely they are to get more of it
- Examples:
  - the more friends you have, the easier it is to make more
  - the more business a firm has, the easier it is to win more
  - the more people there are at a nightclub, the more who want to go
- Such processes will amplify inequality
- One simple and general model: if you have amount \( x \) of something, the probability you get more is proportional to \( x \)
  - so if you have twice as much as me, you’re twice as likely to get more
- Generally leads to heavy-tailed distributions (power laws)
Preferential Attachment

• Start with two vertices connected by an edge
• At each step, add one new vertex v with one edge back to previous vertices
• Probability a previously added vertex u receives the new edge from v is proportional to the (current) degree of u
  – more precisely, probability u gets the edge is 
    \[ \frac{\text{current degree of } u}{\text{sum of all current degrees}} \]
• Vertices with high degree are likely to get even more links!
  – just like Instagram, Twitter, …
• Generates a power law distribution of degrees
• Variation: each new vertex initially gets d edges
Barabasi-Albert (BA) model

• The BA model for preferential attachment
  – input: some initial subgraph $G_0$, and $d$ the number of edges per new node
  – the process:
    • nodes arrive one at the time
    • each node connects to $d$ other nodes selecting them with probability proportional to their degree
    • if $[d_1,\ldots,d_t]$ is the degree sequence at time $t$, the node $t+1$ links to node $i$ with probability equal to
      \[
      \frac{d_i}{\sum_i d_i}
      \]
  • Guarantees a degeneracy of $d$. Why?
  • Brute-force algorithm runs in $O(n^2)$ time. (Bad.)
Faster Barabási-Albert (BA) Algorithm

• Let $d$ be the parameter for the BA algorithm

**ALG. 5:** preferential attachment

**Input:** number of vertices $n$
minimun degree $d \geq 1$

**Output:** scale-free multigraph

$G = (\{0, \ldots, n-1\}, E)$

$M$: array of length $2nd$  

// $M$ is an array of edges chosen so far.

for $v = 0, \ldots, n-1$ do

  for $i = 0, \ldots, d-1$ do

    $M[2(vd+i)] \leftarrow v$

draw $r \in \{0, \ldots, 2(vd+i)\}$ uniformly at random

    $M[2(vd+i)+1] \leftarrow M[r]$

$E \leftarrow \emptyset$

for $i = 0, \ldots, nd-1$ do

  $E \leftarrow E \cup \{M[2i], M[2i+1]\}$
Barabasi-Albert (BA) algorithm

- Faster algorithm runs in $O(nd) = O(n+m)$ time.
- The BA model should result in power-law degree distribution with exponent $c = -3$.

$c = -3$. different d’s. $P(k)$ changes. $c$ does not.