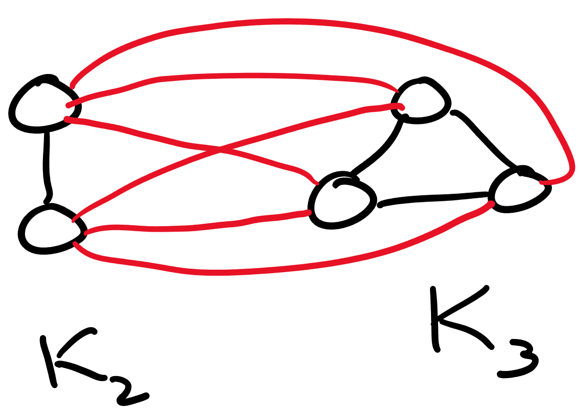
Hamiltonian paths → spanning paths

Recall: graph join between G and H, notationally $G \vee H$, is a construction where $\forall u \in V(G), \forall v \in V(H)$ we add edge (u,v)

join K_2 and K_3 → $K_2 \vee K_3$

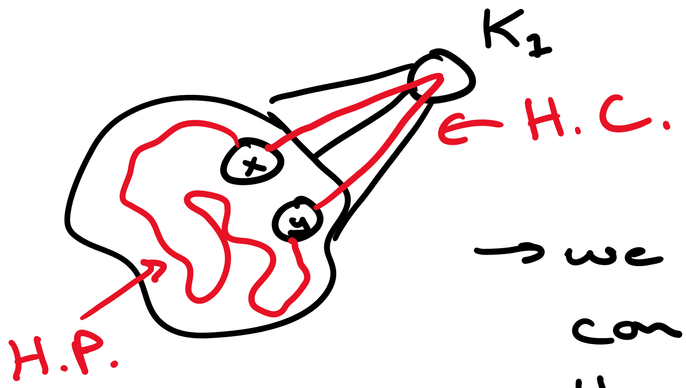


$K_2 \vee K_3 \cong K_5$

G has a Hamiltonian path iff $G \vee K_1$ has a

Hamiltonian cycle

(⇒)



→ we can trivially construct a Hamiltonian cycle via K_1 and start|end of path

(⇐) Necessarily, exactly 2 edges of the cycle are incident on K_1 → delete then and a H.P. remains on G □

When considering the above, we can modify Chvátal's Condition for Hamiltonian paths on G (really just checking for cycle on $G \vee K_1$)

If G has degrees

If G has degrees

$d_1 \le d_2 \le \dots \le d_n$

then if

$i < \frac{n+1}{2}$ implies $d_i \ge i$

or $d_{n+1-i} \ge n-i$

$\Rightarrow$ G has a Ham. path

Note: also sufficient but

not necessary

RANDOM GRAPHS

Q: What are random graphs?

A: A graph with randomly

configured edges in "some way"

(sp?)

configured edges in "some way"
(sp?)

→ Probabilistic models

→ Random generative process

whY?: Mirror properties of real graphs for analytical study

Also: random graphs can be used as null models

How do we define a random graph model?

→ Classic Model

Erdös-Rényi

O.G.: $G(n, m)$ $\quad \langle k \rangle = \frac{2m}{n}$

#verts #edges avg. degree

→ All m edges have randomly selected endpoints chosen selected endpoints chosen from n vertices with replacement

Newer $G(n, p)$

n: # verts

p: attachment probability

↳ prob. that any (u, v) edge exists

→ We consider all possible u, v pairs and create an edge with prob. p (Bernoulli process)

↓

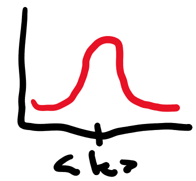
can give us explicit edges or a model for the <u>degree distribution</u>

↳ How many degree-1, degree-2, ..., degree-n vertices?

$$P(k) = \binom{n-1}{k} p^k (1-p)^{(n-1)-k}$$

$P(k)$: prob. of

prob. of
degree k



mean value $\Rightarrow \langle k \rangle = \sum_{k=0}^{n-1} k p_k = p(n-1)$

We can use the above to study graph properties

- Consider vertex v
  - v is expected to have $\langle k \rangle$ neighbors
    $|N(v)| = \langle k \rangle$
  - Each of v's neighbors is also expected to have $\langle k \rangle$ neighbors

→ 2-hop neighborhood of v

$$|N_2(v)| = \underset{N(v)}{\langle k \rangle}\,\underset{N(N(v))}{\langle k \rangle} = \langle k \rangle^2$$

Note: we generally need to make assumptions

(assuming n is suitably large)

(also ignoring that v is in $N(N(v))$)

(

need to make assumptions

$N(N(v))$

$\hookrightarrow$ assume $\langle k \rangle \approx \langle k \rangle - 1$

In general:

$$|N_d(v)| = \langle k \rangle + \langle k \rangle^2 + \langle k \rangle^3 \ldots \langle k \rangle^d$$

$\uparrow$
d-hop neighborhood

$$|N_d(v)| \approx \frac{\langle k \rangle^{d+1} - 1}{\langle k \rangle - 1}$$

consider as $d \to n$

Note: we can take $|N_d(v)| = n$ to determine for what d value the entire graph is contained

(aka, we can get the graph diameter)

$$|N_d(v)| = n \approx \frac{\langle k \rangle^{d+1} - 1}{\langle k \rangle - 1}$$

$$n \approx \langle k \rangle^d$$

$$d \approx \frac{\ln(n)}{\ln(\langle k \rangle)}$$

$$d \approx \frac{\ln(n)}{\ln(\langle k \rangle)}$$

as $\langle k \rangle \ll n$

$\hookrightarrow$ $\boxed{d \approx \ln(n)}$

$\Rightarrow$ expected diameter grows logarithmically with $|V(G)|$

Note: this does generally mirror reality

BIG issue: degree/vertex homogeniety (sp?)

Introducing:

the <u>configuration model</u>

→ this will give us a random graph with some target degree distribution

Basic idea:

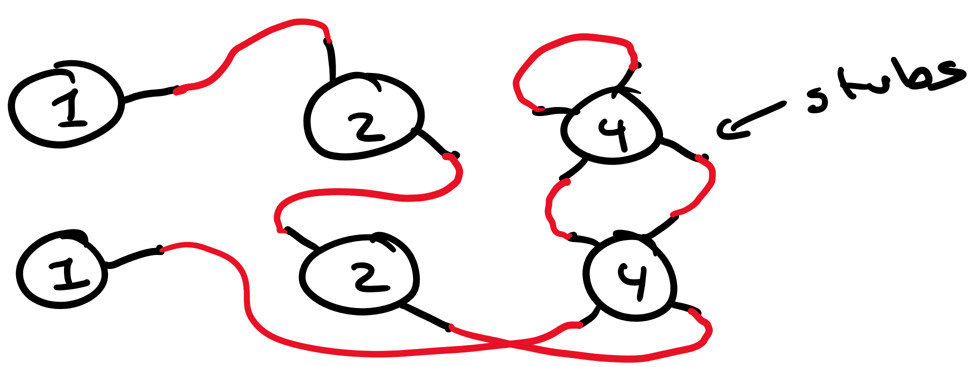
We have some bijection from the degree sequence to a set of verts

→ vertex v has $d(v)$ specified by the degree sequence

→ vertex v has $d(v)$ stubs

→ we randomly wire these stubs together

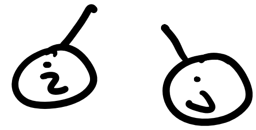
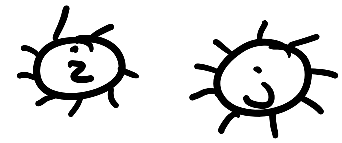
$S = \{4\ 4\ 2\ 2\ 1\ 1\}$



Note: we are generating a loopy multigraph

(random graph models can generate some class of graphs)

What about attachment probs?

Consider  vs. 

Note: more likely to select stubs from high degree vertices

→ an edge is more likely to exist between two high degree vertices vs. two low degree vertices

Consider attachment probs. for some $i, j, d(i), d(j), m$

Prob. of edge (i, j)

= (prob. of selecting i's stub)

*

(prob. of selecting j's stub)

*

2 ← we can select (i,j) or (j, i)

*

$*$

m ← we select m total edges

Prob of i's stub: $\frac{d(i)}{2m}$

$$P_{i,j} = \frac{d(i)}{2m}\,\frac{d(j)}{2\cancel{m}}\,2\cancel{m}$$

i,j attachment probability

$$\boxed{P_{i,j} \approx \frac{d(i)\,d(j)}{2m}}$$

Configuration model

attachment probs

Chung-Lu

$$P_{i,j} = \frac{w_i\, w_j}{\sum_{k=1}^{n} w_k}$$ ← generalization to 'weights' instead of degrees

$$\sum_{k=1} w_k$$

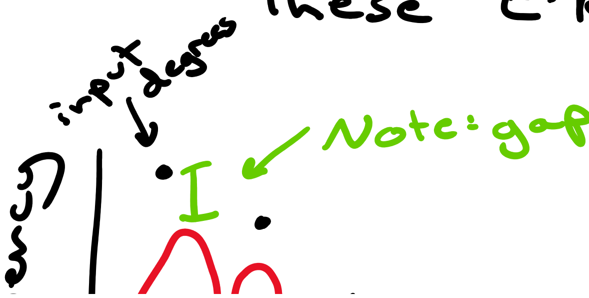
We can generate a simple graph by evaluating $p_{i,j}$ for all i,j pairs

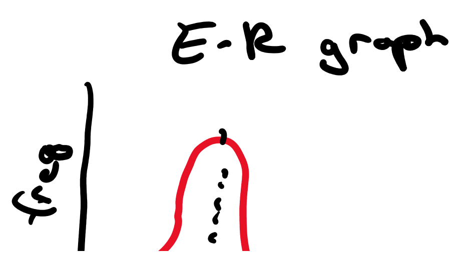
Note: we won't be hitting the degree distributions exactly

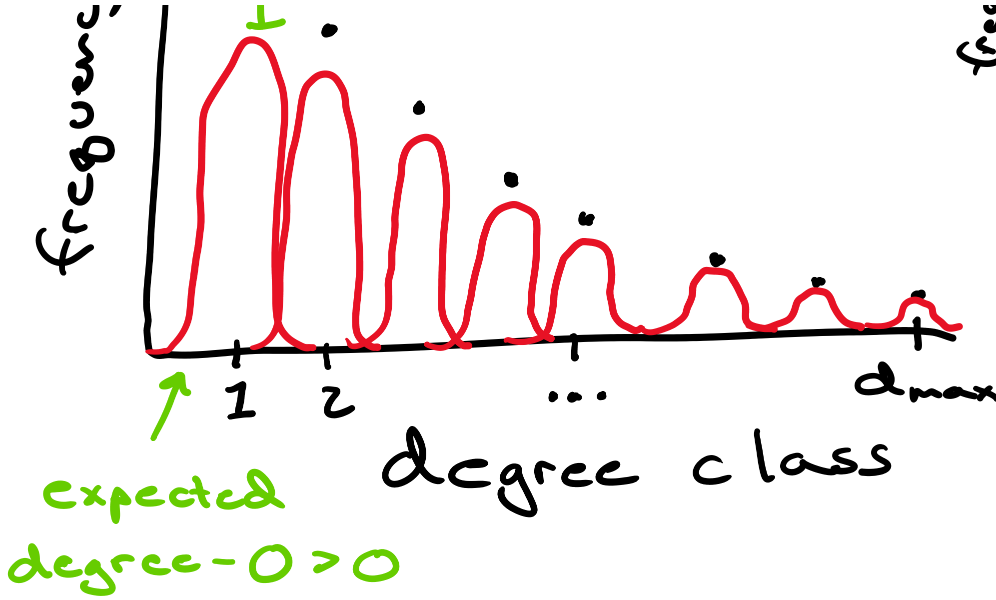
(in expectation we are (not really))

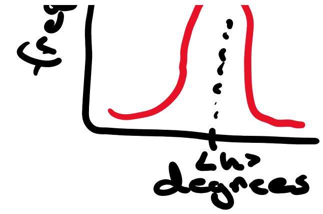
Note x2: we're really just layering a bunch of Erdös-Rényi $G(n,p)$ graphs on top of each other

→ a vertex's degree is a summation of degrees resulting from all of these E-R graphs









One more issue: $p_{i,j} = \frac{d(i)d(j)}{2m}$

↳ what happens when $d(i)d(j) > 2m$

Can happen when a graph is dense or when a DD is skewed

multi-graph → p_{ij} is just the expected # of edges between i,j

simple → nonsense

Null models

→ a graph defined with some properties uniformly randomly selected from all possible graph topologies with those properties

→ n, m, DD, etc.
degree dist.

For some n, m → Erdös-Rényi

For loopy multi-graphs with some DD → configuration model

For simple graphs with some DD → ??

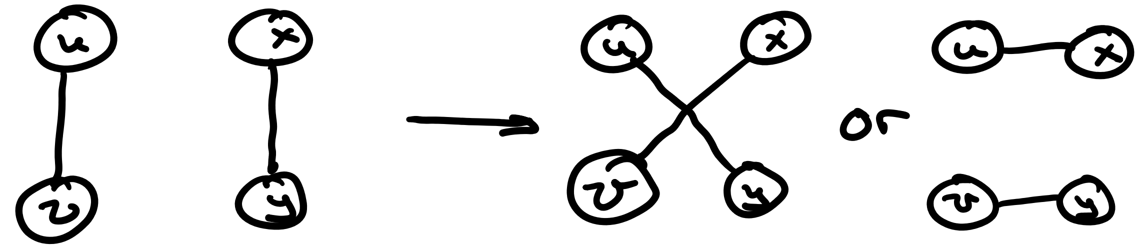
Chung-Lu probs. are biased

→ How can we get an unbiased sample?

A1: not via attachment probs.

A2: via double-edge swaps (and Havel-Hakimi)

Double-edge swap



Note: degrees are unmodified

An approach for simple null model generation:

- Generate H-H graph
- Perform "some number" of double-edge swaps (not allowing loops/multi-edges)

→ this traverses the entire graph class under consideration

Q: how long to remove bias?

A: who knows?

↳ open problem: "mixing time"

To actually get "real" attachment

To actually get "real" attachment probabilities

→ repeat the above "some number" of times, measuring how often degree class i connects to degree class j

Motif finding

what subgraphs appear more often than "expected"

↳ relative to null model

Usually a functional reason