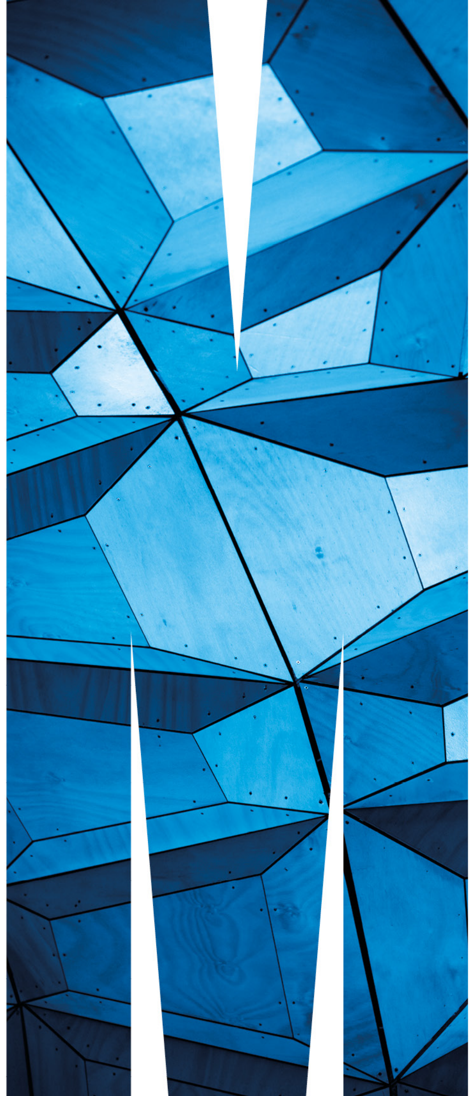
# Spreading Processes on Networks: Models, Techniques and Algorithms

Hans De Sterck hans.desterck@monash.edu
School of Mathematical Sciences
Monash University, Melbourne, Australia
(formerly University of Waterloo)



## “numerical techniques for social media and network problems”

- “computational social science”:
  - large-scale data is being collected on complex social systems
    - online social networks (Facebook, Twitter, ...)
    - email (gmail, ...)
    - travel patterns (public transportation, google maps, ...)
    - mobile phone connections, locations
    - shopping patterns
    - ...
  - it is now possible to build, analyze, and simulate computational models of these systems
  - research is possible that applies successful methods from the natural sciences, e.g. mathematical modelling and statistical mechanics, to produce novel insights in the social sciences → apply "scientific method" to social science
  - some of this can be high-performance computing / high-end computing (big data, combinatorial, ...), efficient algorithms, methods

MONASH University

# “numerical techniques for social media and network problems”

- I have started to set some steps in computational social science / network problems / social media (numerical PDEs, numerical linear algebra, numerical optimization, HPC)
  - multigrid methods for computing stationary vectors of Markov chains – random walks on graphs (Google PageRank) ➔ Nelly Litvak
  - multilevel co-clustering for social networks
  - location tagging for Twitter messages
  - optimization methods for tensor decomposition, matrix completion, recommendation
  - ODE and network models for social uprisings (Arab Spring)
  - dynamical models for smoking epidemic and obesity epidemic
  - propagation of Susceptible-Infectious-Recovered (SIR) disease on random networks with spatial structure

# “numerical techniques for social media and network problems”

“propagation of Susceptible-Infectious-Recovered (SIR) disease on random networks with spatial structure”

- “introduction to network science”
- some new results on SIR propagation on random spatial networks

- Spreading Processes on Networks:
  - Models
  - Techniques
  - Algorithms (random network generation, stochastic simulation algorithms)
  - Applications

Random Spatial Networks: Small Worlds without Clustering, Traveling Waves, and Hop-and-Spread Disease Dynamics

John Lang, Hans De Sterck, Jamieson L. Kaiser, Joel C. Miller

arXiv:1702.01252v1

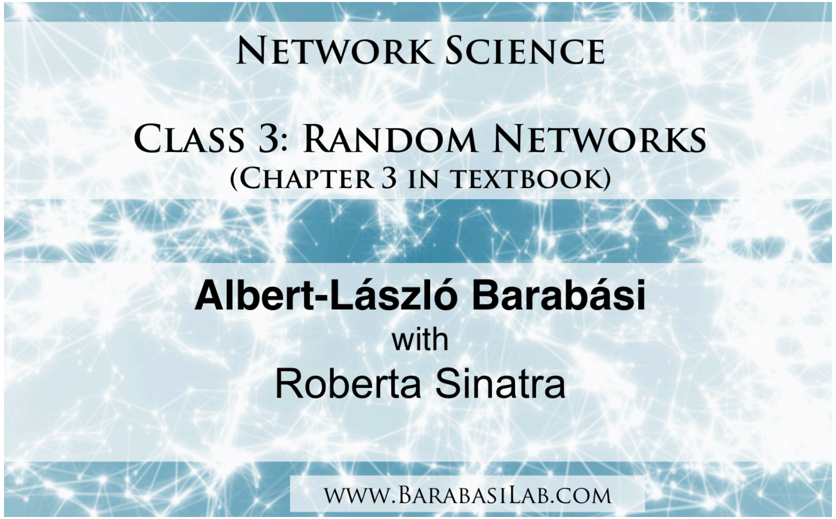
# collaborators

- Joel Miller, Institute for Disease Modeling, Seattle, USA

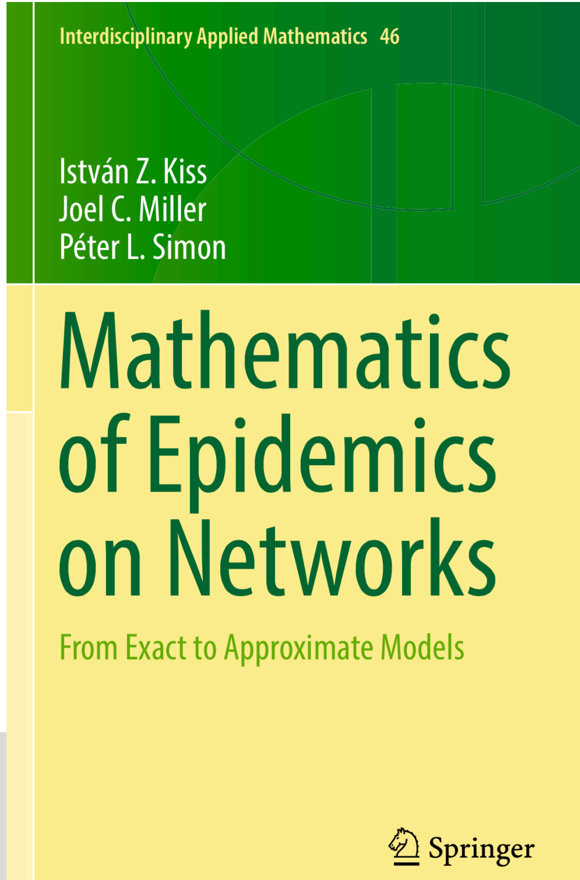
- John Lang, UCLA, Communications Studies (PhD Waterloo, July 2016)

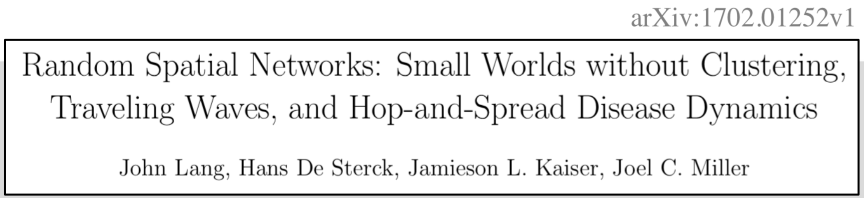
## sources (background on network science, SIR disease propagation, ...)

- http://barabasi.com/networksciencebook/




- book by Kiss, Miller and Simon (2017)





arXiv:1702.01252v1

Random Spatial Networks: Small Worlds without Clustering, Traveling Waves, and Hop-and-Spread Disease Dynamics

John Lang, Hans De Sterck, Jamieson L. Kaiser, Joel C. Miller

# motivation: spread of 2013-2016 Ebola epidemic

- Guinea, Sierra Leone, Liberia

**Virus genomes reveal factors that spread and sustained the Ebola epidemic**

Gytis Dudas, Luiz Max Carvalho, Trevor Bedford, Andrew J. Tatem, Guy Baele,

*Nature* **544**, 309–315 (20 April 2017) | doi:10.1038/nature22040

- goal: develop modeling framework
- random networks
- spatial structure!
- disease propagation (stochastic, DEs for insight)

# two parts of my presentation

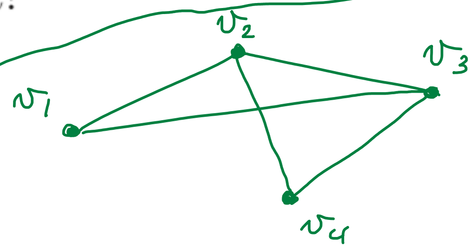
- part A: models and algorithms for networks
  – "introduction to network science"
  – random spatial networks
  – algorithms for efficient network generation
  – application: small worlds with spatial structure

- part B: disease propagation on networks
  – propagation of Susceptible-Infectious-Recovered (SIR) disease
  – stochastic simulation algorithms
  – exact analytic models, and simulations
  – applications

# A1: a brief overview of graphs and networks

- graph $G = (V, E)$ (undirected, simple (no loops, no multiple edges))
  - vertices/nodes $V = \{v_1, v_2, \ldots, v_n\}$ $\qquad n = |V|$
  - edges $E = \{\{v_{i_1}, v_{j_1}\}, \ldots, \{v_{i_m}, v_{j_m}\}\}$ $\qquad m = |E|$

- recall binomial coefficients $\binom{n}{k} = \frac{n!}{(n-k)!\,k!} \qquad (n \geq k)$




$n = 4$

$m = 5$

$V = \{v_1, v_2, v_3, v_4\}$

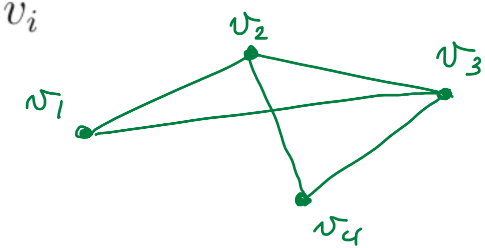
$E = \{\{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_3\},$
$\{v_2, v_4\}, \{v_3, v_4\}\}$

- $m_{\max} = \binom{n}{2} = \frac{n(n-1)}{2}$

$= 6 \quad (n = 4)$

# nodal degrees

- degree $k_i = \text{number of edges incident on node } v_i$
- average degree $< k >= \frac{\sum_{i=1}^{n} k_i}{n}$
- property: $\sum_{i=1}^{n} k_i = 2m$ $\qquad m = \frac{n < k >}{2}$



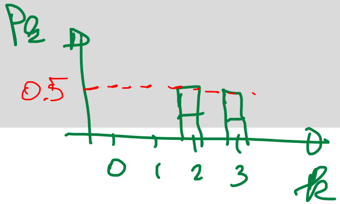

$k_1 = 2$

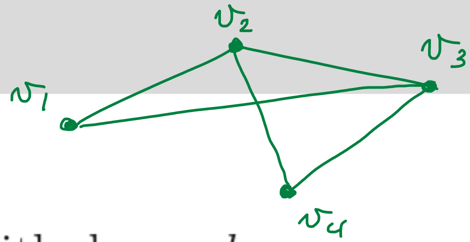
$k_2 = 3$

$< k > = 2.5$

$5 = \frac{4 \cdot 2.5}{2}$

# degree distribution




- degree distribution

$$p_k = \text{fraction of nodes with degree } k$$

- degree distribution for random graph model

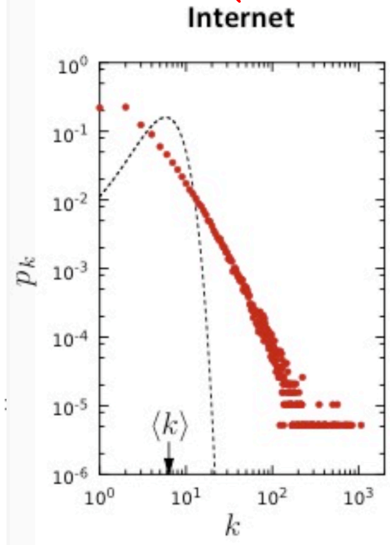
$$P(K = k) = P(\text{a node in the random network has degree } k)$$

- many "real-world" networks approximately have power law degree distribution

$$p_k = ck^{-\gamma} \qquad 2 \leq \gamma \leq 3$$

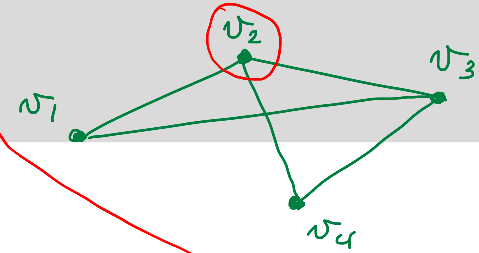
$\ln p_k = \ln c - \gamma \ln k$

Routers




[Barabasi]

# counting triangles - clustering coefficient




- local clustering coefficient of node $v_i$

$$c_i = \frac{\text{\# of triangles node } v_i \text{ forms with its neighbors}}{\text{\# of possible triangles node } v_i \text{ can form with its neighbors}}$$

$c_2 = \frac{2}{3}$

$c_1 = \frac{1}{1}$

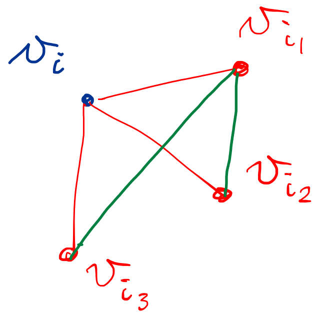
- neighbor set of node $v_i$

$k_i$ neighbors

$$N_i = \{v_{i_1}, \ldots, v_{i_{k_i}}\} \qquad E_{N_i} = \{\text{edges between neighbors of node } v_i\}$$

then

$$c_i = \frac{|E_{N_i}|}{\binom{k_i}{2}} = \frac{2|E_{N_i}|}{k_i(k_i-1)}$$

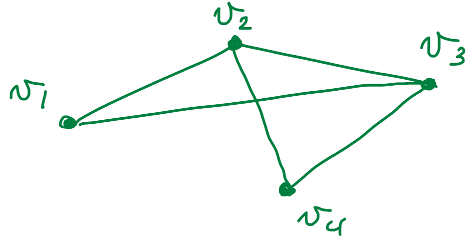



$\langle c \rangle = \frac{1+1+2/3+2/3}{4} = \frac{10}{12} = \frac{5}{6}$

- average clustering coefficient $\quad \langle c \rangle = \frac{\sum_{i=1}^{n} c_i}{n}$

# average shortest path

- $$d_{ij} = \# \text{ of edges between nodes } v_i \text{ and } v_j\text{, in a shortest path}$$




- average (shortest) path length (in a connected graph)

$$< d > = \frac{\sum_{i>j} d_{ij}}{n(n-1)/2}$$

| | |
|---|---|
| 1 – 2 | 1 |
| 1 – 3 | 1 |
| 1 – 4 | 2 |
| 2 – 3 | 1 |
| 2 – 4 | 1 |
| 3 – 4 | 1 |

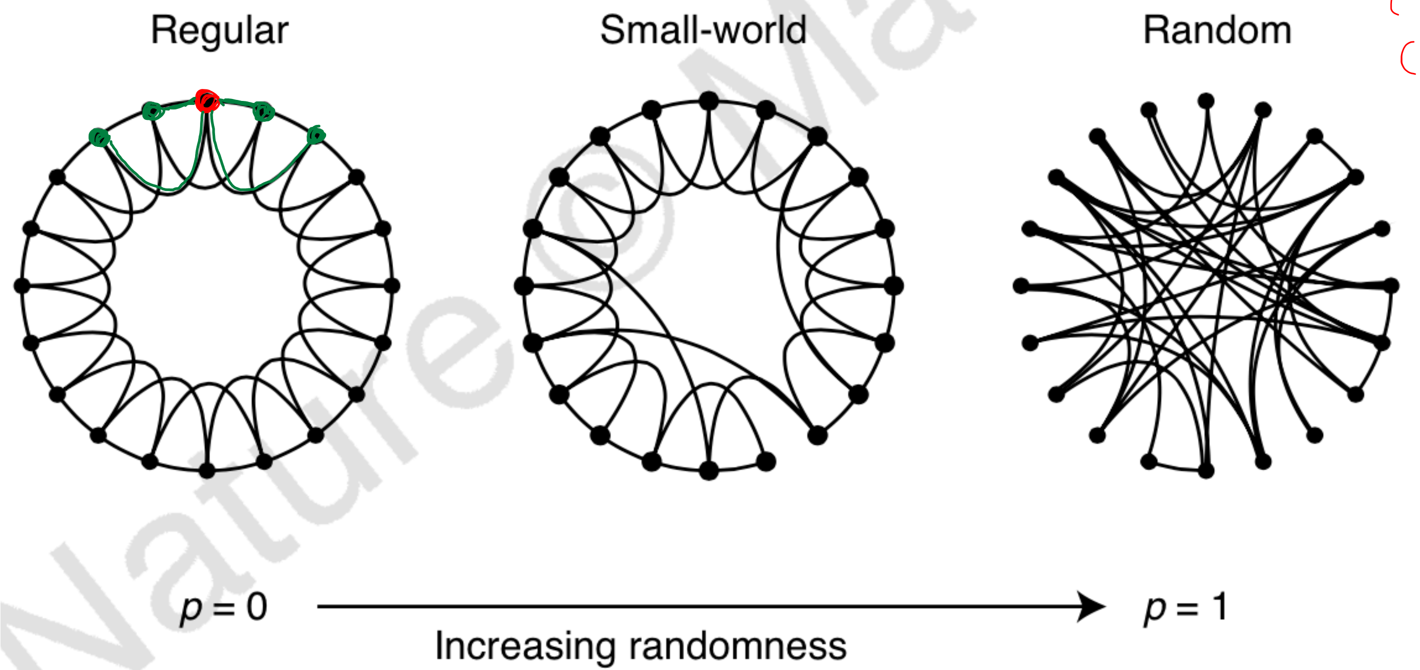
$$<d> = \frac{7}{6}$$

- many “real-world networks” have
  - large clustering $< c >$:
  - small average path length $< d >$

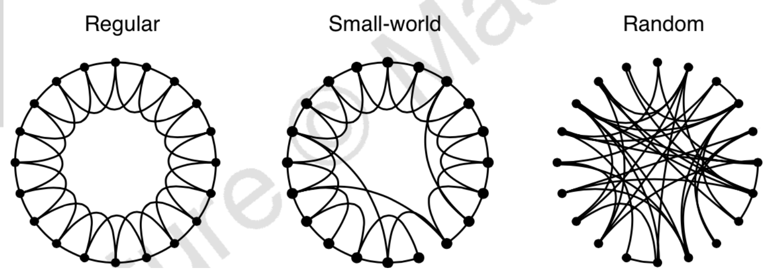
## example: Watts-Strogatz "small world" networks


- ring, each node connected to four nearest neighbors; randomly rewire with probability p

(Nature)
(1998)
(34,000 citations)



# example: Watts-Strogatz “small world” networks



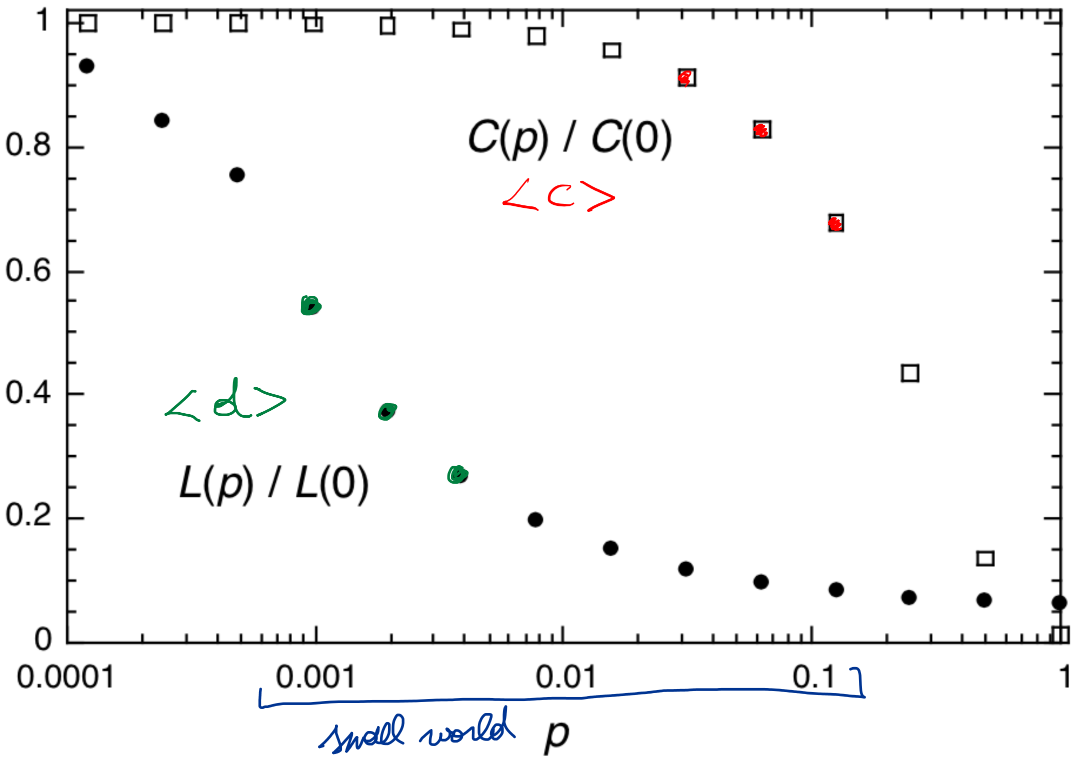

- small world:
  - high clustering $\langle c \rangle$
    (local structure) (unlike random graph)
  - small average (shortest) path length $\langle d \rangle$
    (good connectivity) (like random graph)

“6 degrees of separation”

- many “real-world” graphs are small-world

“large world”: (e.g., lattice)
- local structure
- large (average) shortest path length

$\langle d \rangle = O(n)$
1D lattice

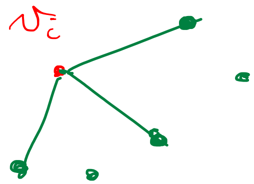


# A2: some random network models

- Erdos-Renyi networks: *n* nodes, and assign edges randomly with probability *p* $G(n,p)$

$$p_{ij} = P(\text{edge } \{v_i, v_j\} \text{ exists})$$

$$p_{ij} = p$$




  - degrees and edges:

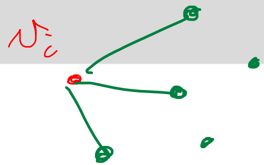
$$k_i \approx (n-1)p$$

$$E(k_i) = (n-1)p$$

$$E(< k >) = (n-1)p$$

$$E(m) = \frac{n(n-1)}{2}p$$

# Erdos-Renyi networks: degree distribution




- binomial degree distribution

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

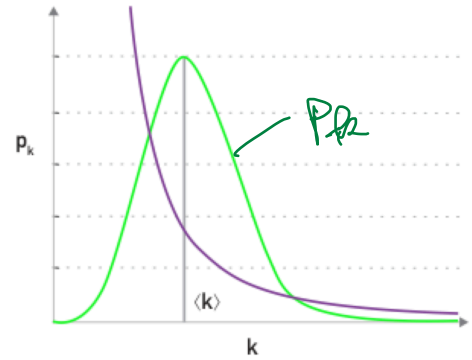
$$E(k) = (n-1)p \qquad \mathrm{Var}(k) = \sigma^2(k) = p(1-p)(n-1)$$

- peaked distribution!

$$\frac{\sigma(k)}{E(k)} = O\left(\frac{1}{\sqrt{n-1}}\right)$$

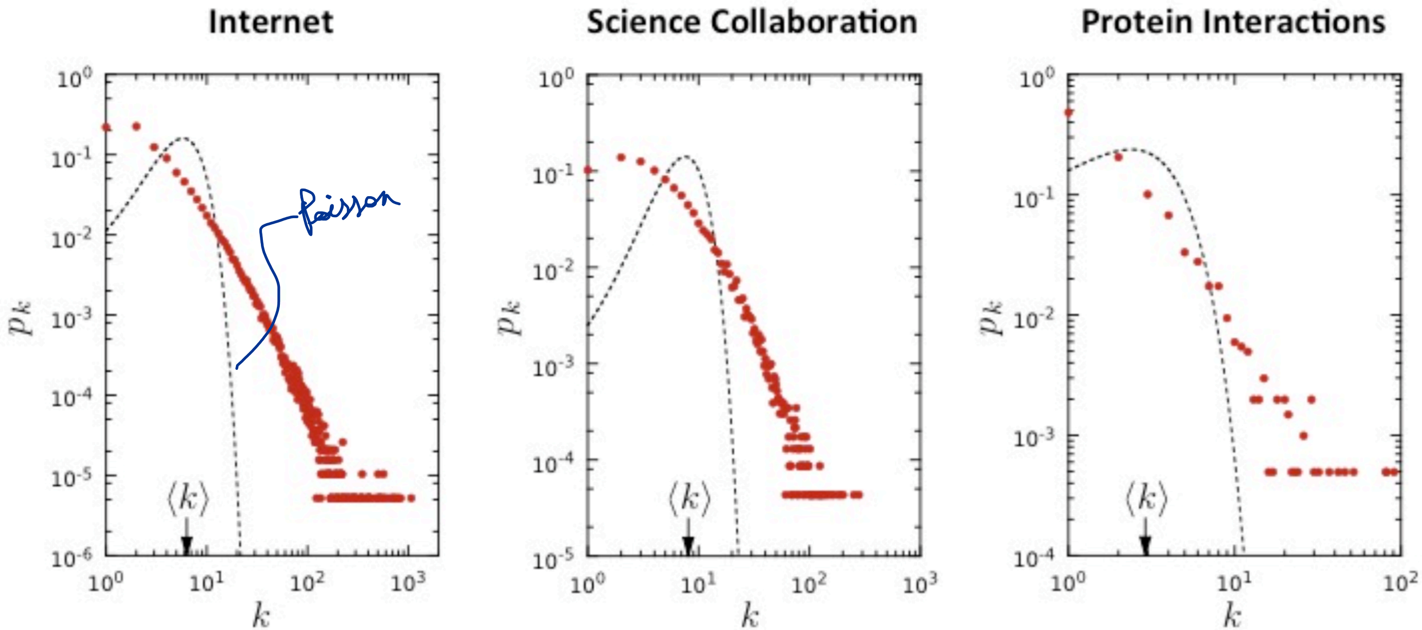
- for large n, small k: approximately Poisson distributed

$$p_k \approx \exp(-\lambda)\frac{\lambda^k}{k!} \qquad E(k) = \lambda \qquad \lambda = (n-1)p$$




[Barabasi]

## degree distributions of “real” networks: power law graphs – scale-free networks




$$p_k = ck^{-\gamma}$$

$$2 \leq \gamma \leq 3$$

[Barabasi]

# “real world” networks with power law distribution (approximately)

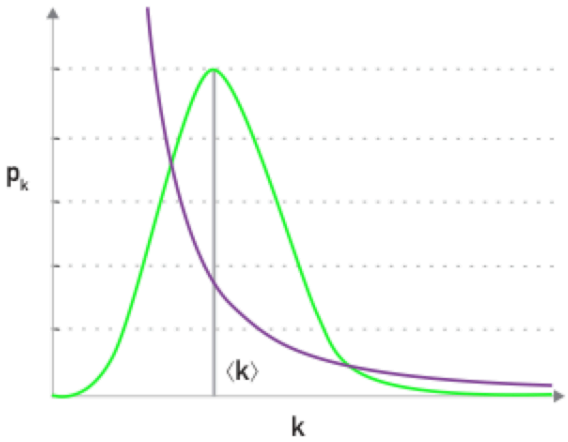
| | Network | $N$ | $L$ | $\langle k \rangle$ | $\langle k_{in}^2 \rangle$ | $\langle k_{out}^2 \rangle$ | $\langle k^2 \rangle$ | $\gamma_{in}$ | $\gamma_{out}$ | $\gamma$ |
|---|---|---|---|---|---|---|---|---|---|---|
| tech | Internet | 192,244 | 609,066 | 6.34 | - | - | 240.1 | - | - | 3.42* |
| | WWW | 325,729 | 1,497,134 | 4.60 | 1546.0 | 482.4 | - | 2.00 | 2.31 | - |
| | Power Grid | 4,941 | 6,594 | 2.67 | - | - | 10.3 | - | - | Exp. |
| social | Mobile-Phone Calls | 36,595 | 91,826 | 2.51 | 12.0 | 11.7 | - | 4.69* | 5.01* | - |
| | Email | 57,194 | 103,731 | 1.81 | 94.7 | 1163.9 | - | 3.43* | 2.03* | - |
| | Science Collaboration | 23,133 | 93,437 | 8.08 | - | - | 178.2 | - | - | 3.35* |
| | Actor Network | 702,388 | 29,397,908 | 83.71 | - | - | 47,353.7 | - | - | 2.12* |
| | Citation Network | 449,673 | 4,689,479 | 10.43 | 971.5 | 198.8 | - | 3.03* | 4.00* | - |
| bio | E. Coli Metabolism | 1,039 | 5,802 | 5.58 | 535.7 | 396.7 | - | 2.43* | 2.90* | - |
| | Protein Interactions | 2,018 | 2,930 | 2.90 | - | - | 32.3 | - | - | 2.89*- |

Table 4.1

[Barabasi]

MONASH University

degree distributions of “real” networks: power law graphs – scale-free networks




Random Network
Randomly chosen node: $k = \langle k \rangle \pm \langle k \rangle^{1/2}$
Scale: $\langle k \rangle$

Scale-Free Network
Randomly chosen node: $k = \langle k \rangle \pm \infty$
Scale: none

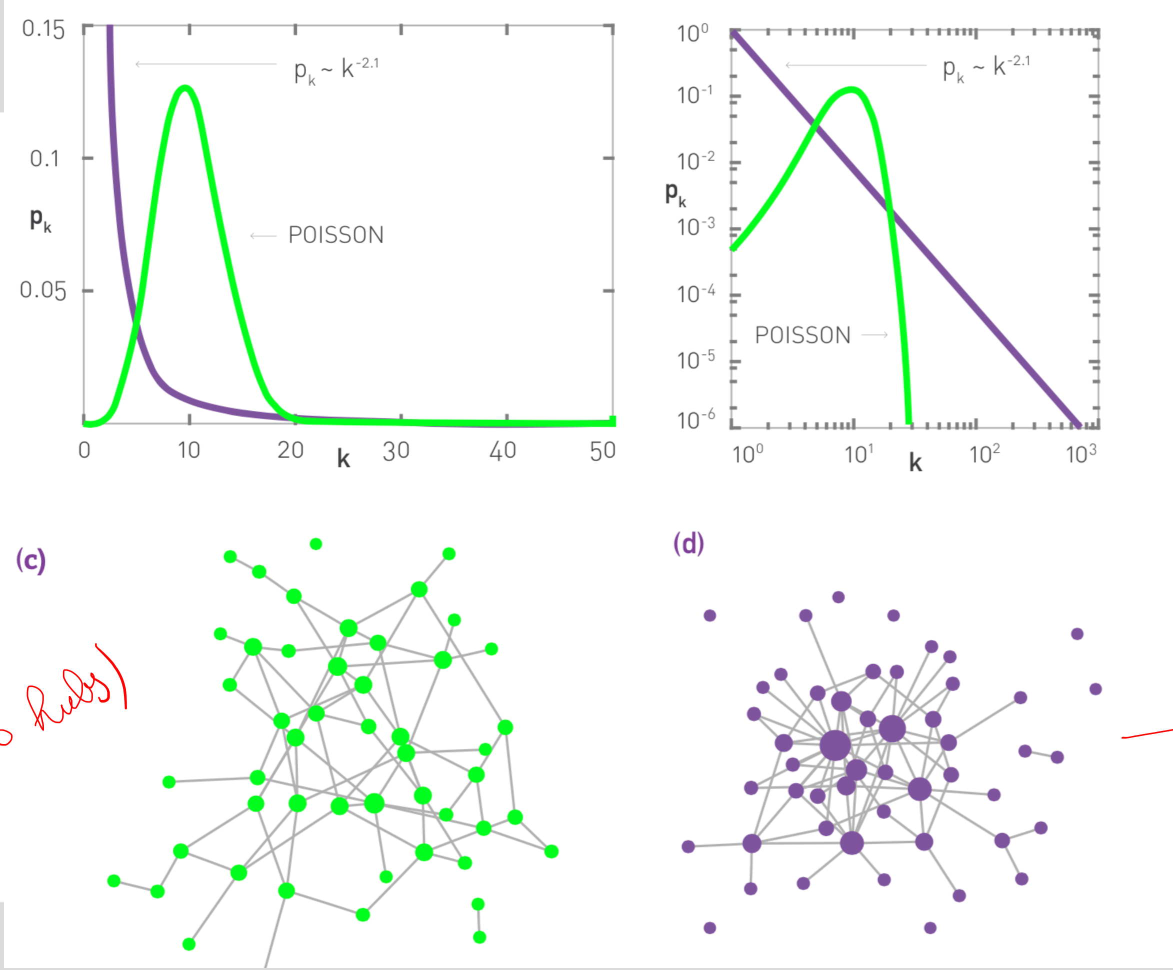
[Barabasi]

$$p_k = ck^{-\gamma} \qquad 2 \leq \gamma \leq 3$$

$$E(k^l) \qquad l > \gamma - 1$$ diverges

$$\gamma < 3 : l = 2, 3, \ldots \ \text{diverge}$$

$$E(k^2)$$ diverges



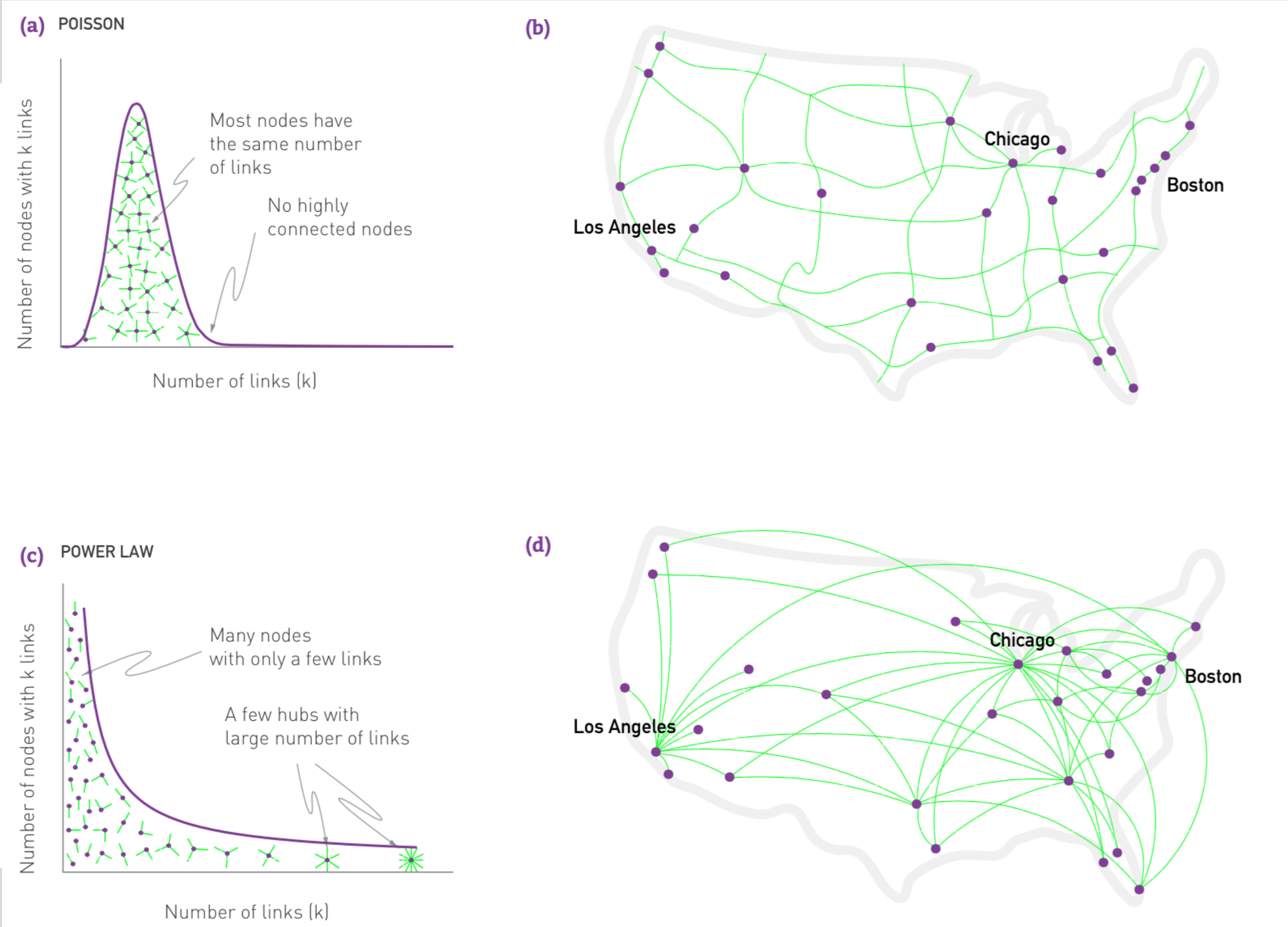

(no hubs)

hubs

[Barabasi]

road network

air network



# Erdos-Renyi networks: average shortest path

- small-world network!

$$d_{ij} = \#\text{ of edges between nodes } v_i \text{ and } v_j\text{, in a shortest path}$$

$$E(d) \approx \frac{\log(n)}{\log(E(k))}$$

average shortest path: $E(d) = O(\log(n))$, or slower

- compare
  - 1D lattice: $E(d) = O(n)$
  - 2D lattice; $E(d) = O(\sqrt{n})$

large worlds

# “real-world” networks: average shortest path

diameter

- small worlds!

$$E(d) \approx \frac{\log(n)}{\log(E(k))}$$

Erdos – Renyi

| Network | N | L | ‹k› | ‹d› | $d_{max}$ | ln*N*/ln‹*k*› |
|---|---|---|---|---|---|---|
| Internet | 192,244 | 609,066 | 6.34 | 6.98 | 26 | 6.58 |
| WWW | 325,729 | 1,497,134 | 4.60 | 11.27 | 93 | 8.31 |
| Power Grid | 4,941 | 6,594 | 2.67 | 18.99 | 46 | 8.66 |
| Mobile-Phone Calls | 36,595 | 91,826 | 2.51 | 11.72 | 39 | 11.42 |
| Email | 57,194 | 103,731 | 1.81 | 5.88 | 18 | 18.4 |
| Science Collaboration | 23,133 | 93,437 | 8.08 | 5.35 | 15 | 4.81 |
| Actor Network | 702,388 | 29,397,908 | 83.71 | 3.91 | 14 | 3.04 |
| Citation Network | 449,673 | 4,707,958 | 10.43 | 11.21 | 42 | 5.55 |
| E. Coli Metabolism | 1,039 | 5,802 | 5.58 | 2.98 | 8 | 4.04 |
| Protein Interactions | 2,018 | 2,930 | 2.90 | 5.61 | 14 | 7.14 |

Table 3.2

[Barabasi]

MONASH University

# Erdos-Renyi networks: clustering coefficient

$$c_i = \frac{2|E_{N_i}|}{k_i(k_i-1)} \approx p = \frac{E(k)}{n-1}$$

$$\frac{\text{connected neighbor pairs}}{\text{total neighbor pairs}}$$

- low clustering as n grows

- but: many “real-world” networks have high average clustering coefficients

| Network | $C$ | Erdös-Rényi |
|---|---|---|
| Web [2] | 0.081 | 7.71 |
| Flickr | 0.313 | 47,200 |
| LiveJournal | 0.330 | 119,000 |
| Orkut | 0.171 | 7,240 |
| YouTube | 0.136 | 36,900 |

(A. Bonato)

## random networks with more realistic degree distributions: Chung-Lu networks

- $n$ nodes, desired degree sequence $\{\kappa_1, \ldots, \kappa_n\}$
- add edges randomly according to

$$p_{ij} = P(\text{edge } \{v_i, v_j\} \text{ exists})$$

$$p_{ij} = \frac{\kappa_i \kappa_j}{n < \kappa >} \qquad < \kappa > = \frac{\sum_{i=1}^{n} \kappa_i}{n}$$

$$p_{ij} = \max\left(\frac{\kappa_i \kappa_j}{n < \kappa >}, 1\right)$$

(advantage: edges assigned independently, which keeps the model “analyzable” ... see later)

## Chung-Lu networks: expected degree

- assume $\frac{\kappa_i \kappa_j}{n < \kappa >} \leq 1$

$$
\begin{aligned}
E(k_i) &= \sum_{j=1, j\neq i}^{n} p_{ij} \\
&= \sum_{j=1, j\neq i}^{n} \frac{\kappa_i \kappa_j}{n < \kappa >} \\
&= \sum_{j=1}^{n} \frac{\kappa_i \kappa_j}{n < \kappa >} - \frac{\kappa_i^2}{n < \kappa >} \\
&= \kappa_i - \frac{\kappa_i^2}{n < \kappa >} \\
&\approx \kappa_i
\end{aligned}
$$

- Chung-Lu as a model for “real-world” networks:
  - path length $< d >$
    OK!
  - degree distribution $p_k$
    OK!
  - clustering coefficient $< c >$
    too low ...

MONASH University

# Chung-Lu networks: choose desired degrees from distribution

- note: desired degree sequence $\{\kappa_1, \dots, \kappa_n\}$ can be chosen from (continuous) distribution $p(\kappa)$

$$E(\kappa) = \int_0^\infty \kappa p(\kappa) d\kappa \qquad p_{ij} = \frac{\kappa_i \kappa_j}{n < \kappa >}$$

$$E(k_i) = \int_0^\infty \frac{\kappa_i \kappa}{n \int_0^\infty \kappa p(\kappa) d\kappa} n p(\kappa) d\kappa = \kappa_i$$

if node $v_i$ is assigned desired degree $\kappa_i$,
its expected degree is $\kappa_i$

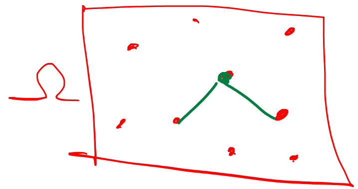
# A3: random spatial networks (RSNs)

Random Spatial Networks: Small Worlds without Clustering, Traveling Waves, and Hop-and-Spread Disease Dynamics

John Lang, Hans De Sterck, Jamieson L. Kaiser, Joel C. Miller

- Chung-Lu:

$$p_{ij} = \frac{\kappa_i \kappa_j}{n < \kappa >}$$

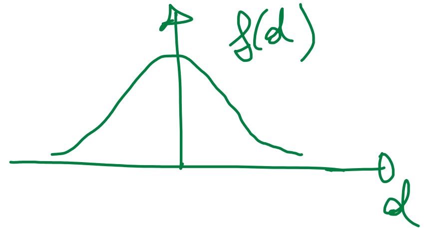



- with spatial structure: in domain $\Omega$

$$p_{ij} = \frac{\kappa_i \kappa_j}{n < \kappa >} \, |\Omega| \; f(d_{ij}) \qquad d_{ij} = \|\vec{x}_i - \vec{x}_j\|_2$$

nodes are placed randomly in Ω

$$\int_{\Omega} f(\|\vec{x}_0 - \vec{x}\|_2) d\vec{x} = 1 \quad \forall \vec{x}_0 \in \Omega \qquad \overline{f} \; |\Omega| = 1$$




- more generally:

$$p_{ij} = \min\left( \frac{\kappa_i \kappa_j}{\rho < \kappa >} f(d_{ij}), 1 \right) \qquad \rho = \frac{n}{|\Omega|}$$

# random spatial networks (RSNs)

$$p_{ij} = \min\left(\frac{\kappa_i \kappa_j}{\rho < \kappa >} f(d_{ij}), 1\right)$$

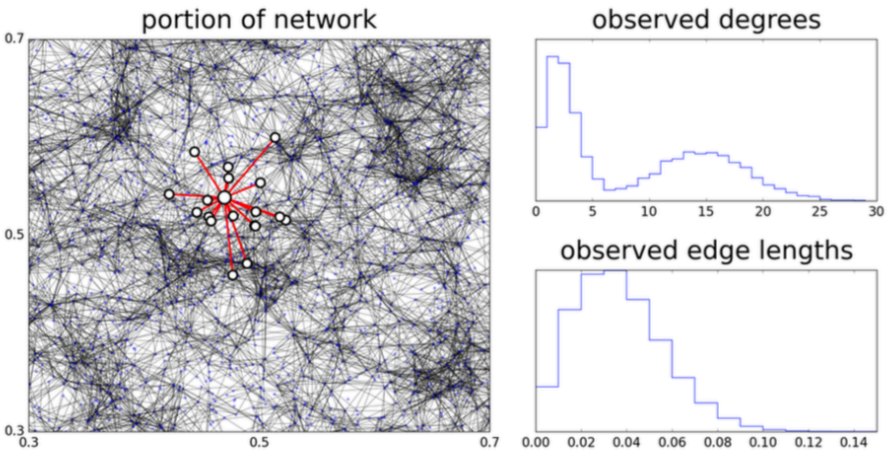



FIG. 1. An example RSN and its properties. The distance kernel is a Gaussian, $f(d) = \exp(-d^2/2\sigma^2)/2\pi\sigma^2$ with $\sigma = 0.03$. The imposed distribution of expected degrees is $P(2) = P(15) = 0.5$. The density is $\rho = 10000$. One node and its neighbors are highlighted. A random network without spatial structure would exhibit neighbors throughout the domain.

$$p_{ij} = \min\left(\frac{\kappa_i \kappa_j}{\rho < \kappa >} f(d_{ij}), 1\right)$$

$\rho = \frac{n}{\Omega}$

$$E(k_i) = \int_{\kappa=0}^{\infty} \left( \int_{\Omega} \frac{\kappa_i \kappa}{\rho \int_0^{\infty} \kappa p(\kappa) d\kappa} f(\|\vec{x}_i - \vec{x}\|_2) \rho d\vec{x} \right) p(\kappa) d\kappa$$

$$= \kappa_i \frac{\int_0^{\infty} \kappa p(\kappa) d\kappa}{\int_0^{\infty} \kappa p(\kappa) d\kappa} \underbrace{\int_{\Omega} f(\|\vec{x}_i - \vec{x}\|_2) d\vec{x}}_{=1}$$

$$= \kappa_i$$

— if node $v_i$ is assigned desired degree $\kappa_i$, its expected degree is $\kappa_i$

- Erdos-Renyi: $G(n, p)$

  $p_{ij} = p$

- naive algorithm
  - for each possible edge $\{v_i, v_j\}$ , draw a random number

$$m_{\max} = \binom{n}{2} = \frac{n(n-1)}{2}$$

  - complexity $O(n^2)$

# efficient algorithms for network generation

**Efficient Generation of Large Random Networks***

Vladimir Batagelj†
*Department of Mathematics, University of Ljubljana, Slovenia.*

Ulrik Brandes‡
*Department of Computer & Information Science, University of Konstanz, Germany.*

- Erdos-Renyi: O(n+m) algorithm $\quad p_{ij} = p$
  - naive algorithm has many failures (when p is small)
  - idea: we know the distribution of successes and failures! (geometric); so sample the number of failures according to the appropriate distribution

$$\delta = \text{ number of failures before success}$$

$$\begin{aligned} P(\delta) &= P(\text{first success happens at trial } \delta + 1) \qquad \delta = 0, 1, 2, \ldots \\ &= (1-p)^{\delta} p \end{aligned}$$

$$\begin{aligned} P(D \leq \delta) &= P(\text{first success happens in trial } 1, 2, \ldots, \text{ or } \delta + 1) \\ &= 1 - P(\text{no success in first } \delta + 1 \text{ trials}) \\ &= 1 - (1-p)^{\delta+1} \end{aligned}$$

MONASH University

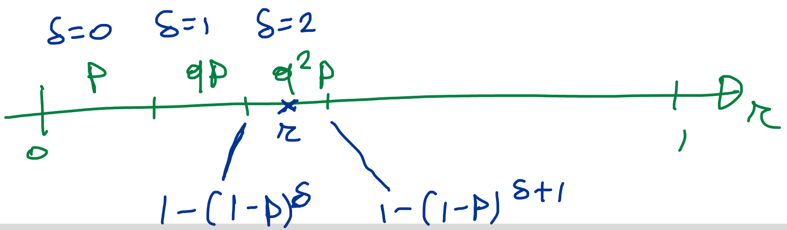
- Erdos-Renyi: O(n+m) algorithm $\quad p_{ij} = p \quad\quad q = 1 - p$

$$\delta = \text{ number of failures before success}$$

$$\begin{aligned} P(D \le \delta) &= P(\text{first success happens in trial } 1, 2, \ldots, \text{ or } \delta + 1) \\ &= 1 - P(\text{no success in first } \delta + 1 \text{ trials}) \\ &= 1 - (1-p)^{\delta+1} \end{aligned}$$

- choose $r \in [0, 1]$

← sample the number of failures




$$\text{choose } \delta \text{ if } 1 - (1-p)^{\delta} \le r \le 1 - (1-p)^{\delta+1}$$

$$(1-p)^{\delta} \ge 1 - r \ge (1-p)^{\delta+1}$$

$$\delta \le \frac{\ln(1-r)}{ln(1-p)} \le \delta + 1$$

$$\delta = \left\lfloor \frac{\ln(1-r)}{ln(1-p)} \right\rfloor$$

**Algorithm 1.** $G(N, p)$ Graph

**Input:** number of nodes $N$, and probability $0 < p < 1$
**Output:** $G(N, p)$ graph $G(V, E)$ with $V = \{0, \ldots, N-1\}$

```
E ← ∅
for u = 0 to N − 2 do
    v ← u + 1
    while v < N do
        choose r ∈ (0, 1) uniformly at random
        v ← v + ⌊log(r) / log(1−p)⌋
        if v < N then
            E ← E ∪ {u, v}
            v ← v + 1
```

N-1

inner loop

N-1 failures total

m successes total

$O(n+m)$ (random numbers)

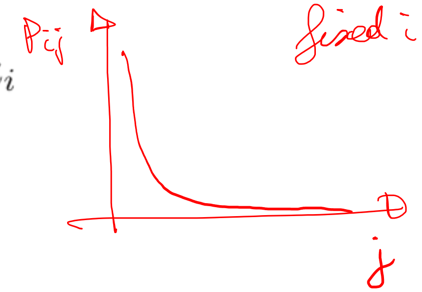
**Efficient Generation of Networks with Given Expected Degrees**

Joel C. Miller[1,2] and Aric Hagberg[3]

- Chung-Lu: O(n+m) algorithm $p_{ij} = \frac{\kappa_i \kappa_j}{\sum_{l=1}^{n} \kappa_l}$

"skip" many edges

fixed i




$v_1$
$v_2$
$v_3$
⋮
⋮
$v_n$

  - order nodes $v_i$ in order of decreasing desired degree $\kappa_i$
  - for fixed *i*:
    - fix $p = p_{i,i+1}$
    - determine number of failures $\delta$ as before (using p)

      ↳ $\delta$ was underestimated, because p was too large
    - $v_{i+1+\delta}$ is a potential neighbor: accept with probability $\frac{q}{p} \leq 1$ where $q = p_{il} \leq p$

      (then $p \frac{q}{p} = q = p_{il}$ ) ("rejection sampling")

      $l = i + 1 + \delta$
    - set $p = q$, compute next $\delta$ (repeat...)
  - acceptance remains high! (increasingly large jumps) $O(n+m)$ can be shown

increasingly large jumps

**Algorithm 2.** Chung-Lu Graph

**Input:** list of $N$ weights, $W = w_0, \ldots, w_{N-1}$, sorted in decreasing order
**Output:** Chung-Lu graph $G(V, E)$ with $V = \{0, \ldots, N-1\}$

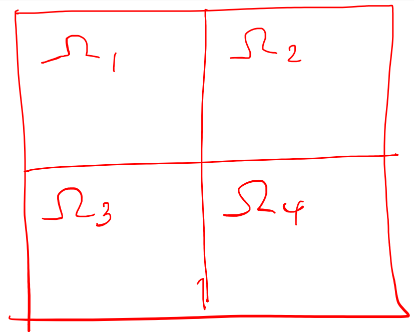
```
E ← ∅
S ← Σ_u w_u
for u = 0 to N − 2 do
    v ← u + 1
    p ← min(w_u w_v / S, 1)
    while v < N and p > 0 do
        if p ≠ 1 then
            choose r ∈ (0, 1) uniformly at random
            v ← v + ⌊ log(r) / log(1−p) ⌋
        if v < N then
            q ← min(w_u w_v / S, 1)
            choose r ∈ (0, 1) uniformly at random
            if r < q/p then
                E ← E ∪ {u, v}
            p ← q
            v ← v + 1
```

if $p = 1$ : $S = 0$ (nothing added)

$O(n+m)$ can be shown (acceptance is kept high)

# efficient algorithms for network generation

- $\boxed{\text{Random Spatial Networks (RSNs):}}$ O(n+m) algorithm




$$p_{ij} = \min\left(\frac{\kappa_i \kappa_j}{\rho < \kappa >} f(d_{ij}), 1\right)$$

assume $f(d)$ bounded, decreasing in $d$

  – divide $\Omega$ into $\ell$ subdomains $\Omega_j \quad (1 \leq j \leq \ell)$

  – order nodes in each subdomain by decreasing desired degree

$$p = \frac{\kappa_i \kappa_j}{\rho < \kappa >} f_{\max}$$

choose $f_{max}$ such that acceptance is kept high (keep $p$ close to the real $p_{ij}$)

determine number of failures $\delta$ as before (using p)

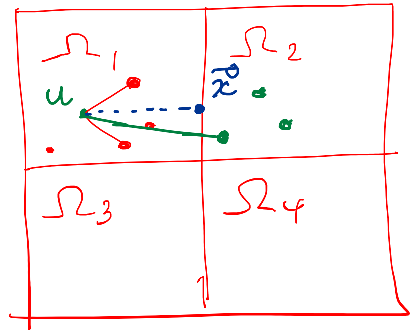
$$p_{ij} = \min\left(\frac{\kappa_i \kappa_j}{\rho < \kappa >} f(d_{ij}), 1\right)$$

- Random Spatial Networks (RSNs): O(n+m) algorithm

$$p = \frac{\kappa_i \kappa_j}{\rho < \kappa >} f_{\max} \longrightarrow \text{make sure } p \geq P_{ij}$$

determine number of failures $\delta$ as before (using p)

node $u$ in region $\Omega_1$




» edges within region 1:

$$f_{\max} = f(0)$$

» edges to region $\Omega_2$

find point $\vec{x}$ in $\Omega_2$ nearest to node $u$ in region $\Omega_1$

$$f_{\max} = f(\|\vec{x} - \vec{x}_u\|_2) \longrightarrow f_{max} \text{ is chosen small, but } p \geq P_{ij}$$

└ high acceptance

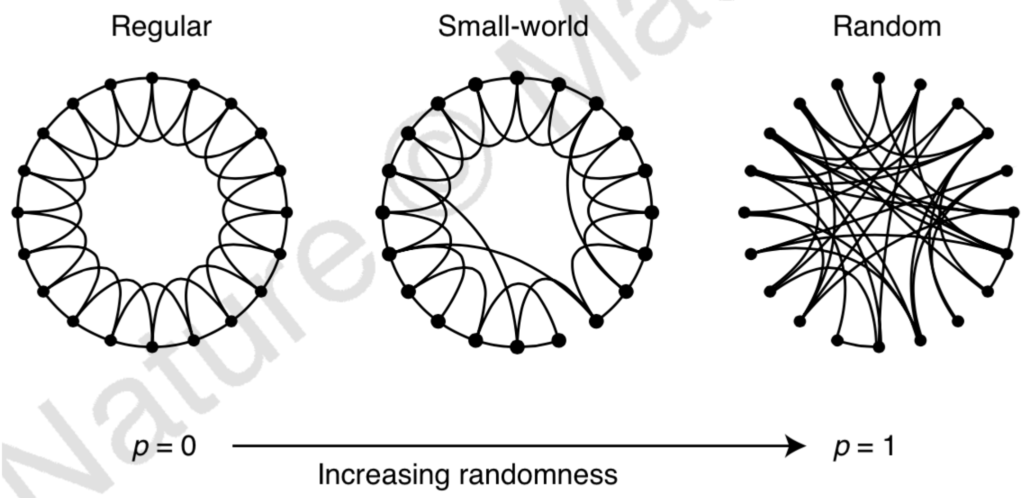
– acceptance remains high! (increasingly large jumps)

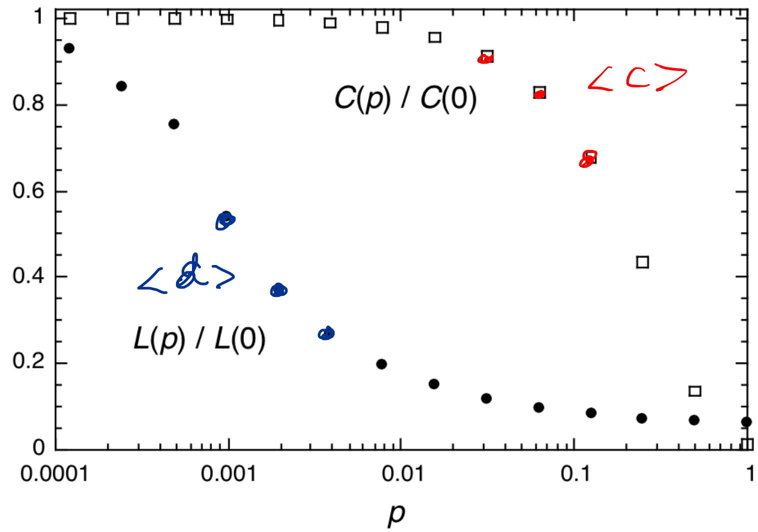
$O(n+m)$

MONASH University

# A5: small world networks with spatial structure (RSNs)

- recall: Watts-Strogatz "small world" networks

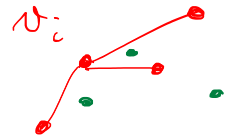



- small world:
  - large average local clustering coefficient $\langle c \rangle$
  - small average (shortest) path length $\langle d \rangle$

# small world networks with spatial structure (RSNs)

- RSNs: define local proximity coefficient
  - first normalize all distances: $\bar{d}_{ij}$ normalized to $[0, 1]$

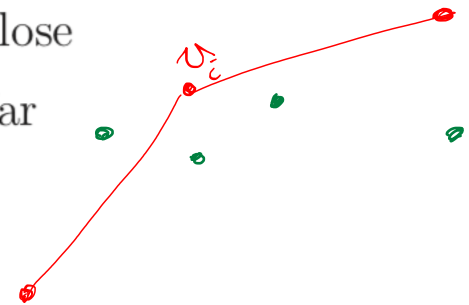



  - define local proximity coefficient:

$$p_i = 1 - \mathrm{avg}(\bar{d}_{ij} \text{ of graph neighbors } j \text{ of } i) \qquad p_i \in [0, 1]$$

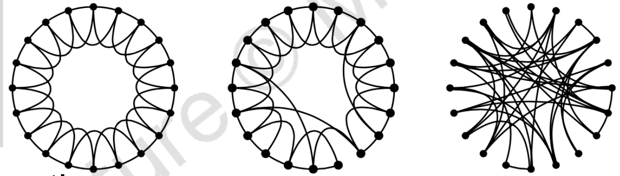
$$p_i \approx 1\text{: graph neighbors of } i \text{ are located close}$$

$$p_i \approx 0\text{: graph neighbors of } i \text{ are located far}$$




  - average local proximity coefficient:
  - 

$$< p >$$



- a specific class of RSNs: mostly local connections, few global connections

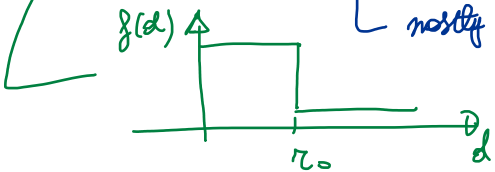
$\kappa_i = k \qquad |\Omega = 1| \qquad$ choose $\; r_0 \qquad\qquad$ compute $\; N_0 = \dfrac{k}{\pi r_0^2}$

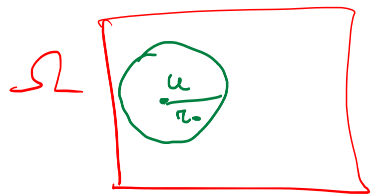
$\Omega^2 / \Omega$

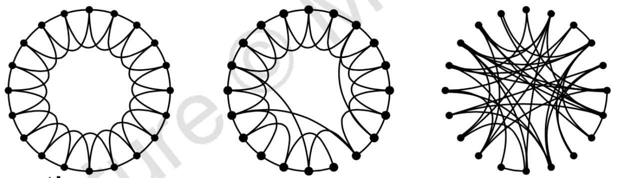
$$p_{ij} = \min\left(\frac{\kappa_i \kappa_j}{\rho < \kappa >} f(d_{ij}), 1\right) \qquad p_{uv} = \min\left(k\frac{f(d_{uv})}{N}, 1\right) = \frac{k f(d_{uv})}{N}$$

$$\boxed{f(d_{uv}) = \begin{cases} \frac{N_0}{k}\left[1 - \epsilon^{\frac{1-\pi r_0^2}{\pi r_0^2}}\right] & d_{uv} < r_0 \\ \frac{N_0}{k}\epsilon & d_{uv} \geq r_0 \end{cases}.} \qquad \boxed{p_{uv} = \begin{cases} \frac{N_0}{N}\left(1 - \epsilon^{\frac{1-\pi r_0^2}{\pi r_0^2}}\right) & d_{uv} < r_0 \\ \frac{N_0}{N}\epsilon & d_{uv} \geq r_0 \end{cases}}$$

mostly local, some global





- a specific class of RSNs: mostly local connections, few global connections

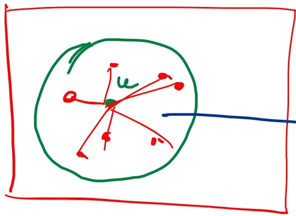
$$f(d_{uv}) = \begin{cases} \frac{N_0}{k}\left[1 - \epsilon\frac{1-\pi r_0^2}{\pi r_0^2}\right] & d_{uv} < r_0 \\ \frac{N_0}{k}\epsilon & d_{uv} \geq r_0 \end{cases}.$$

$$p_{uv} = \begin{cases} \frac{N_0}{N}\left(1 - \epsilon\frac{1-\pi r_0^2}{\pi r_0^2}\right) & d_{uv} < r_0 \\ \frac{N_0}{N}\epsilon & d_{uv} \geq r_0 \end{cases}$$

$$N_0 = \frac{k}{\pi r_0^2}$$

- case $n = N_0,\ \epsilon = 0$ $\quad \frac{k}{N_0} = \frac{\pi r_0^2}{1}$ $\quad < c > \approx 0.59$

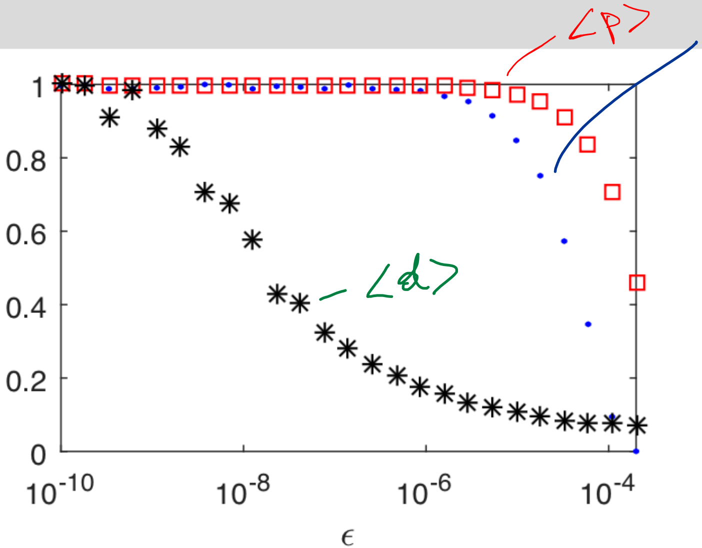
  ⟨ "random geometric graph"

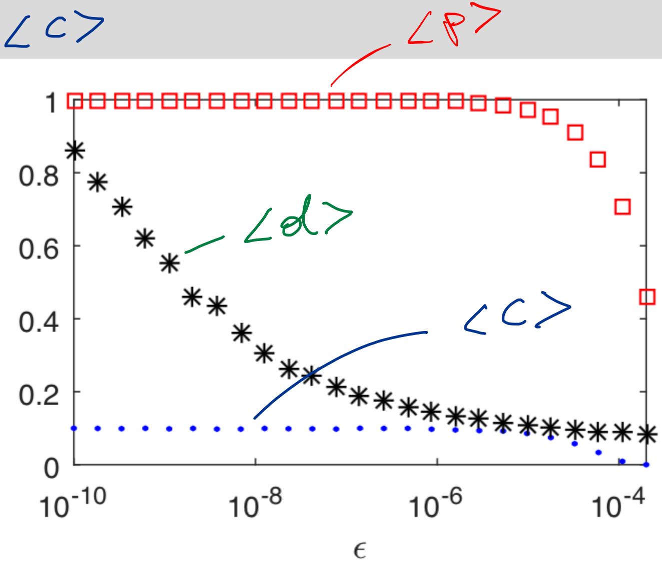


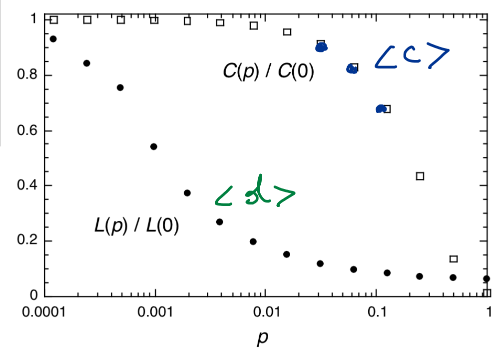

all k neighbors in disc are connected to u

- case $n = N_0,\ \epsilon \neq 0$ : some long-range connections
- case $n = N \gg N_0,\ \epsilon \neq 0$ $\quad < c >= O(1/N)$
  - low density, low clustering, but proximity remains the same

# small worlds with spatial structure

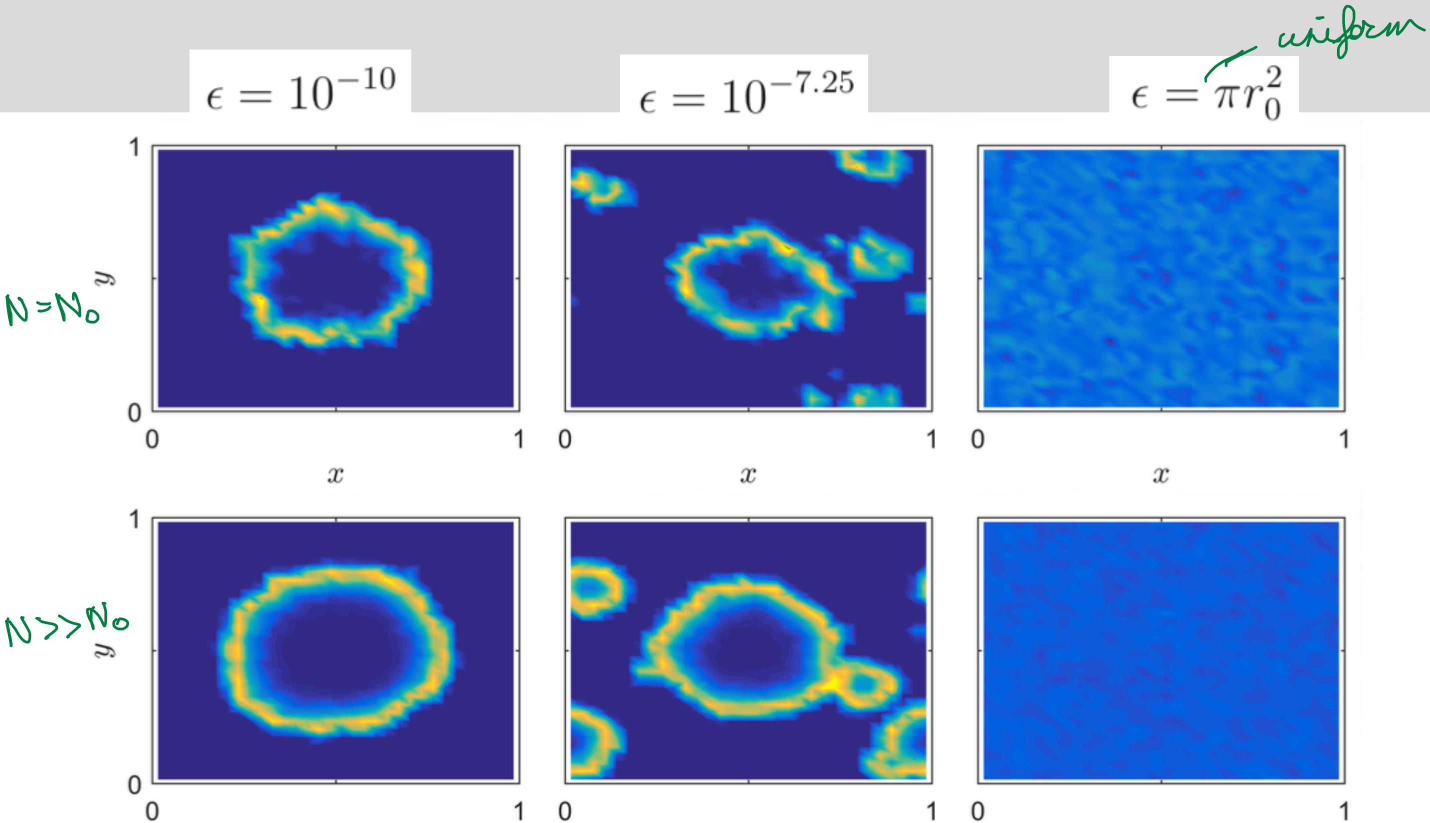



- low density, $N=N_0$: small world
  - high clustering, high proximity
  - low (average) shortest path
- high density, $N>>N_0$: unclustered small world
  - low clustering
  - high proximity : local
  - low (average) shortest path : global

still small world! (local, and global, structure)




big world: local propagation

small world: local and global propagation

uniform world: only global propagation

(no local structure)

small-world effect:

- local propagation (traveling wave) due to local structure (proximity, not clustering)
- long-range jumps due to small-world property

for spatial networks, proximity is important in determining whether small-world effects occur, rather than clustering