Simulations in Statistical Physics

Course for MSc physics students

Janos Török

Department of Theoretical Physics

November 4, 2014

Homework 1

Wolff cluster algorithm Responsible: Gábor Mándi

Write a program that uses the Wolff cluster algorithm for the Ising model on the three-dimensional cubic lattice!

- Determine the spontaneous magnetization and the susceptibility as a function of the temperature and system size.
- Apply finite size scaling to the problem, use small systems.

Homework 2

3D diffusion-limited aggregation. Responsible: György Vida

Write a computer simulation of the two-dimensional diffusion-limited aggregation model on triangular and square lattices: Introduce your own measure of anisotropy parameter and determine the difference between the two lattices. Measure the fractal dimension.

Homework 3

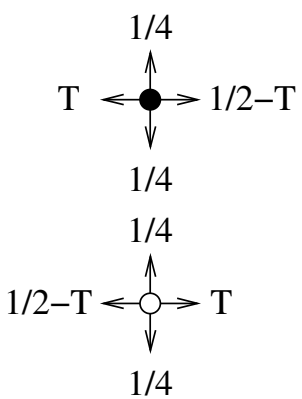
Epidemic model on BA graph Responsible: László Ujfalusi

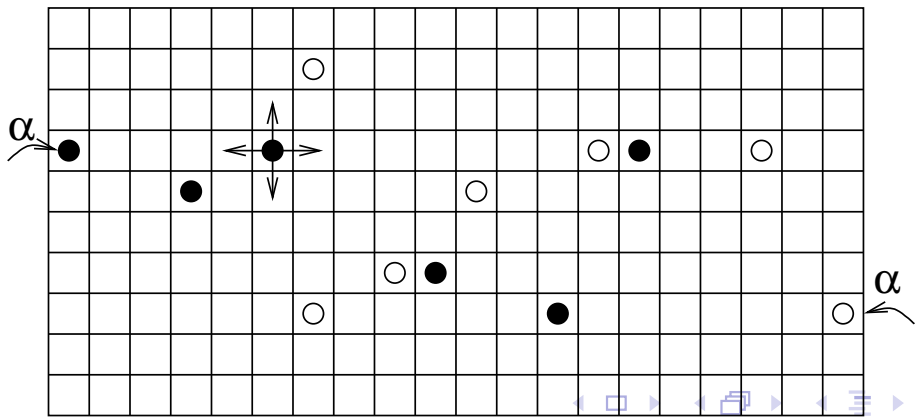
Generate an Erdős-Rényi graph with fixed $\langle k \rangle$. Investigate the following epidemic model on this network:
There are three kinds of nodes: susceptible (**S**), infected (**I**), recovered (**R**). In every time step an infected node infects its susceptible neighbours with probability β, infected sites can recover with probability γ. A recovered site cannot become infected again, and they do not infect any susceptible site.
In the beginning every site is susceptible, except for 10 which are infected. Using the system size $N = 1000$, $\gamma = 0.1$ show that the limit for the infection to grow is $\beta > \langle k \rangle / \gamma$. Use ensemble averages!

Homework 4

Dynamical Monte Carlo simulation Responsible: Balázs Nagyfalusi

Simulate a lattice gas with Monte Carlo method. The lattice is 20x10 sites and is periodic in the vertical direction. There are two type of particles: black and white ones. Black ones are inserted on the left hand side of the lattice with rate α (choose a site, if it is empty put there a black particle with probability α.) White particles are inserted on the right side with the same rate. Particles which leave either left or right the lattice are discarded.
The particles may move to *empty* adjacent sites with the probabilities shown on the figure. Measure (ensemble average) the current and the density in function of α and $T \in [0, 1/2]$.





Homework 5

Page-Rank on BA network Responsible: Grörgy Vida

Create a directed Barabási-Albert network with $m = 2$. Start with a triangle. When a new node is added it has two outgoing links which are attached to nodes with probability proportional to the degree of the node including both the incoming and outgoing links. For $N = 100$ calculate the *page rank* and plot the degree page rank correlation, and age page rank correlation. (Use ensemble average.)

Homework 6

Ising spin glass with genetic algorithm Responsible: Levente Rózsa

Consider an $N \times N$ square lattice with periodic boundary conditions and the Hamiltonian

$$H = -\sum_{<i,j>} J_{ij} s_i s_j .$$

The summation goes over the nearest neighbour pairs. The J_{ij} coupling coefficients are randomized at the start of the simulation by setting them to ± 1 with probability 0.5. The $s_i = \pm 1$ variables represent the Ising spins at the lattice points. Determine the approximate ground state energy per lattice point in the system by using a genetic algorithm, with the set of s_i values on the lattice as the genetic code.

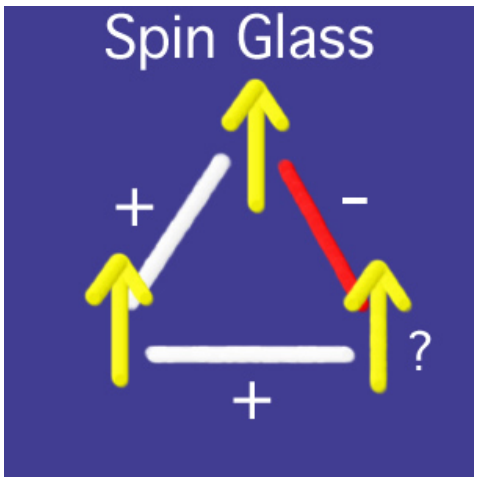
Use a set of 100 different realizations, choose 50 with probability inversely proportional to H. Generate 50 children using random pairs and random gene mixing. Allow for mutations with $p = N^{-2}/2$ probability. Perform the simulation for $N = 8$ and $N = 16$.

Glassy behavior, frustration

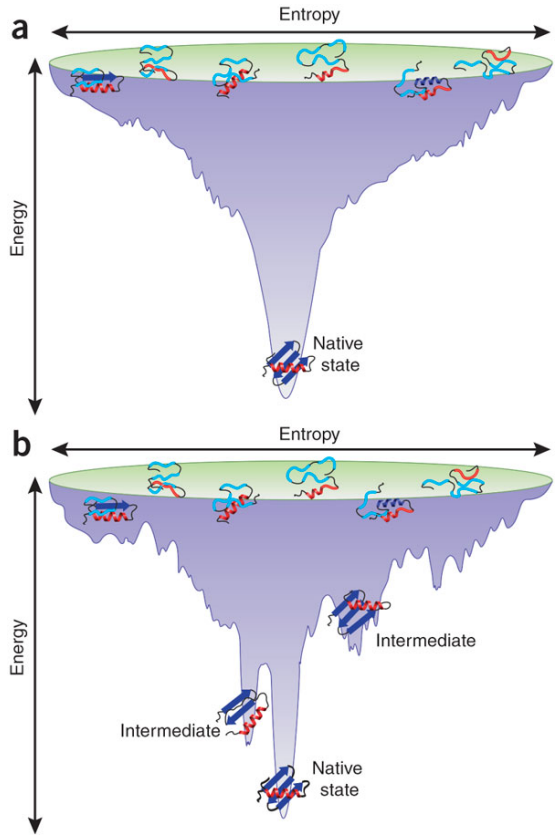
- Model glass: spin-glass:

$$H = -\frac{1}{2}\sum_{\langle i,j\rangle} J_{ij} S_i S_j$$

- where J_{ij} are random quenched variables with 0 mean (e.g. $\pm J$ with probability half)



Rugged energy landscape.



Spin Glasses:memory effects

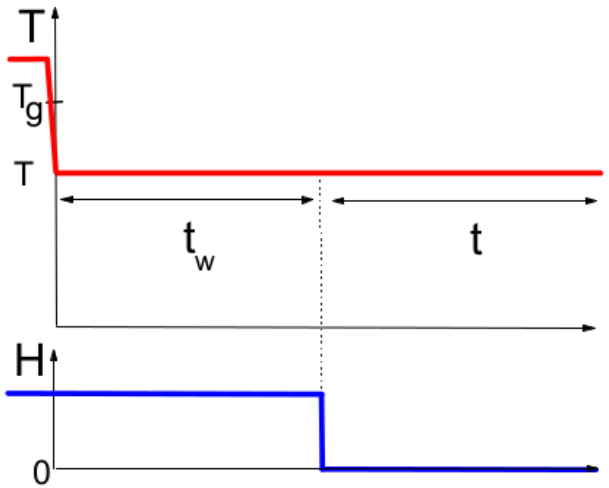
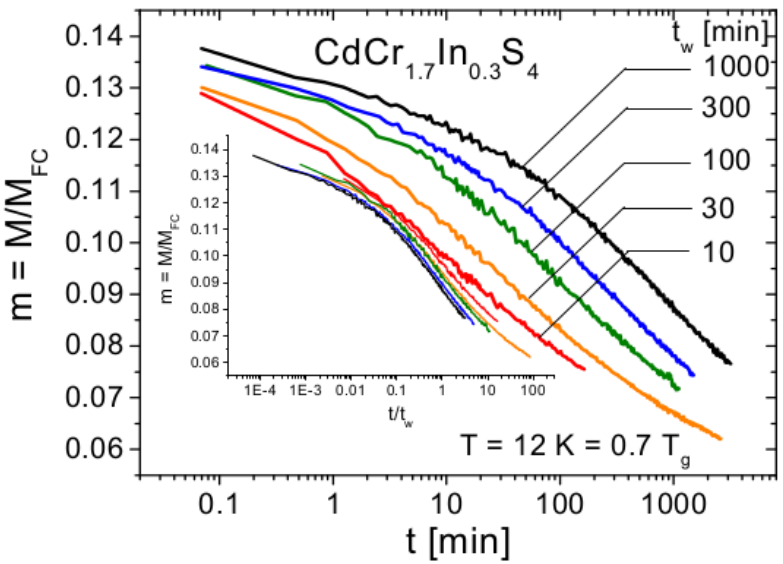
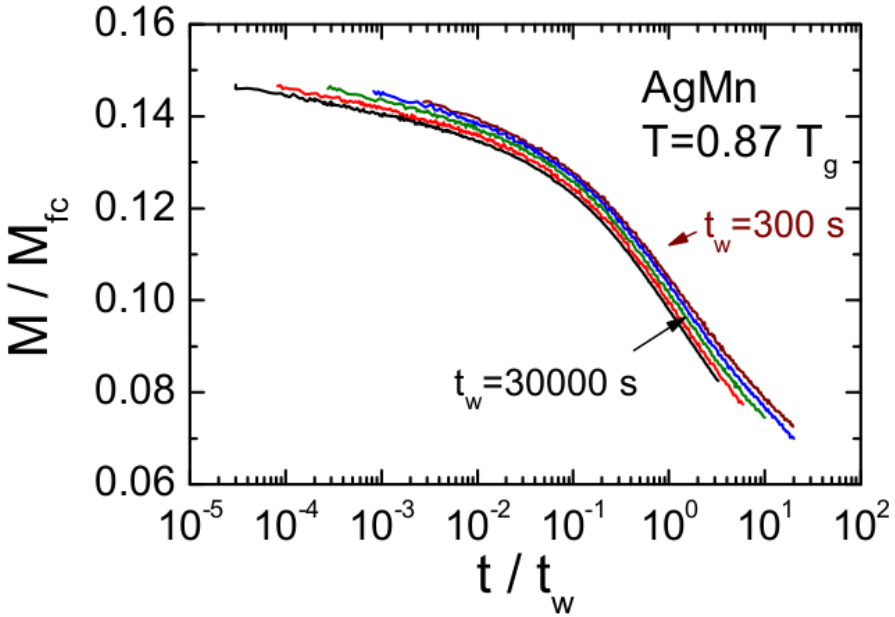
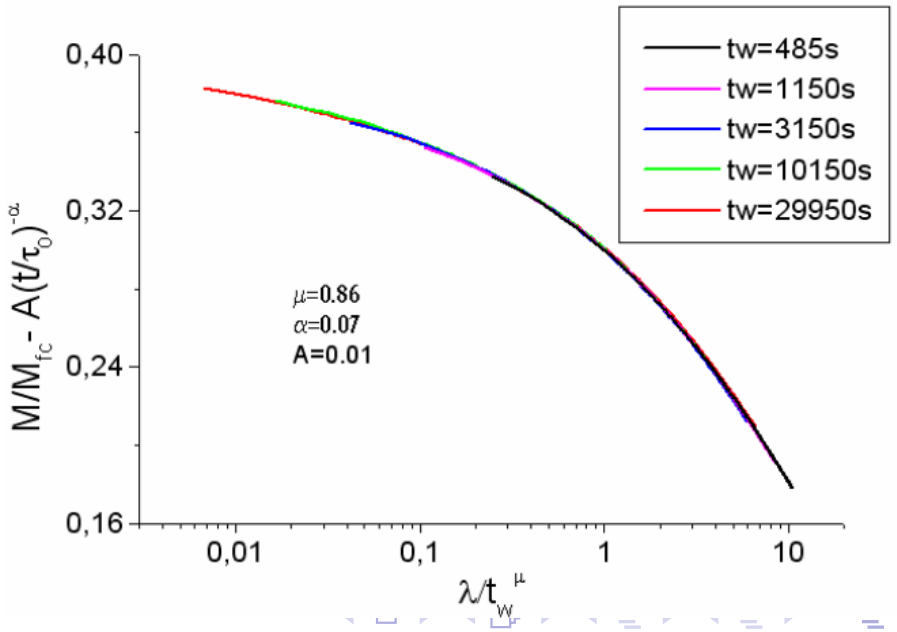


Figure 2.1: Sketch of the TRM measurement procedure.



Spin Glasses:scaling relations



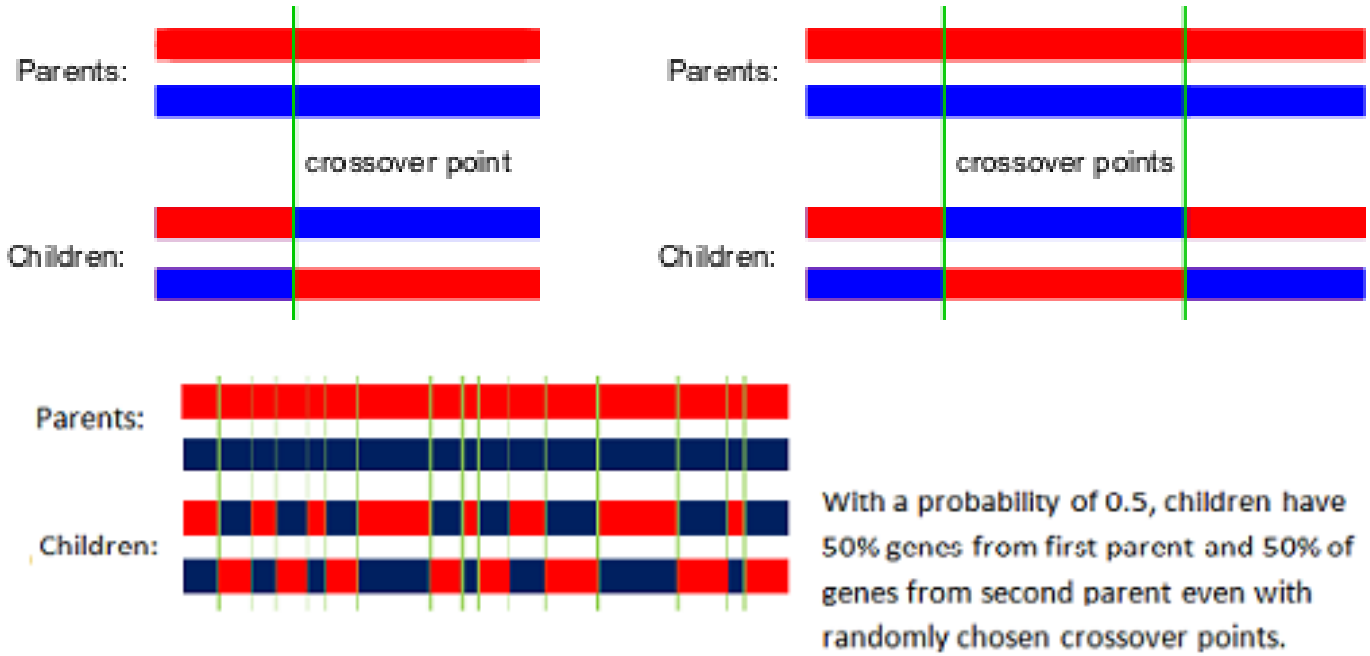


Genetic algorithm

- Learn from nature
- Let the fittest to survive
  - Fitness function, e.g. energy, length, etc.
- Combine different strategies
- State is represented by a vector (genetic code or genotype)
  - Phasespace, city order, neural network parameters, etc.
- Offsprings have two parents with shared genetic code
- Mutations
- Those who are not fit enough die out
  - Keep the number of agents fixed

Genetic algorithm: Reproduction

- Two parents and two children



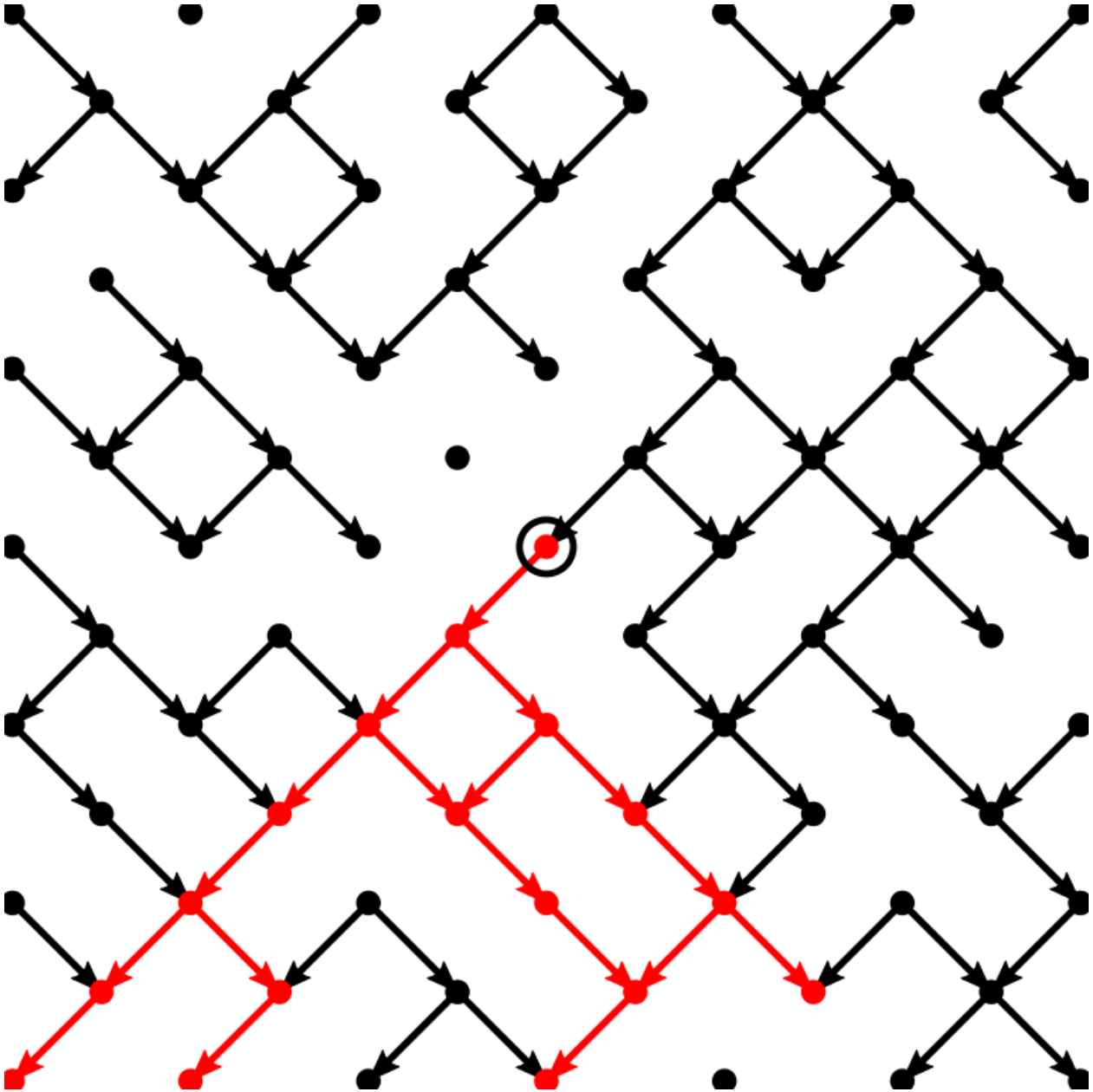
Genetic algorithm terminology

- Chromosome: Carrier of the genetic representation
- Gene: Smallest units in the chromosome with individual meaning
- Parents: Pair of chromosomes, wich produce offsprings
- Population: Set of chromosomes from which the parents are selected. Its size should be larger than the length of the chromosome
- Selection principle: The way parents are selected (random, elitistic)
- Crossover: Recombination of the genes of the parents by mixing
- Crossover rate: The rate by which crossover takes place ($\sim$90%)
- Mutatation: Random change of genes
- Mutation rate: The rate by which mutation takes place ($\sim$1%)
- Generation: The pool after one sweep.

Genetic algorithm

1. Create randomly N agents with chromosomes
2. Calculate the fitness of the agents
3. With probability proportional to the fitness keep some part of the agents (generally half of it)
4. Generate children using two random parents and crossover
5. With probability p_m mutate a gene in each child
6. Go to 2

Directed percolation



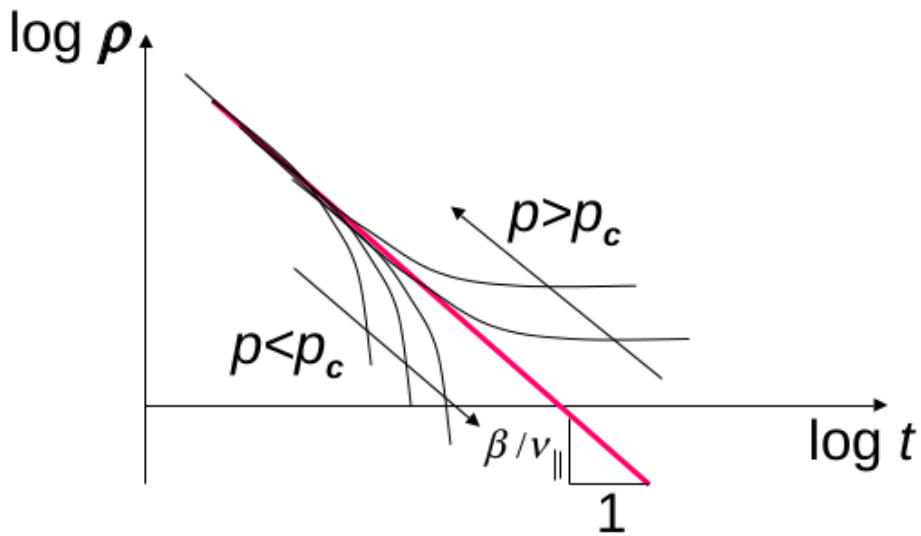
Directed percolation

- More complicated than percolation
- 3 exponents (correlation lengths in two directions) $\nu_\perp$, $\nu_{||}$ and (order parameter) β

$$\rho(\Delta p, t, L) \sim b^{-\beta/\nu_\perp} \rho(b^{1/\nu_\perp} \Delta p, t/b^z, L/b),$$

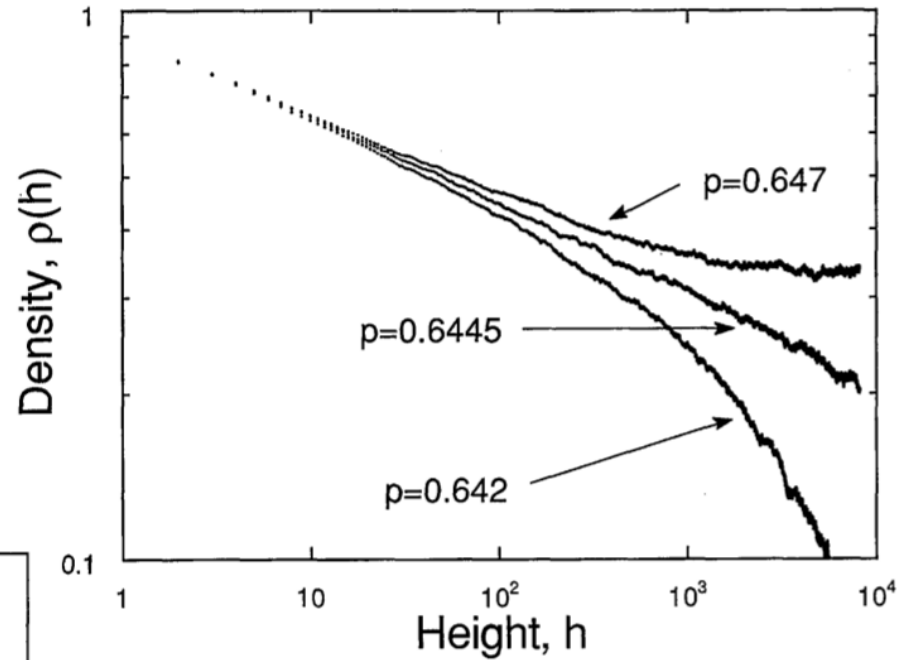
with $z = \nu_{||}/\nu_\perp$.

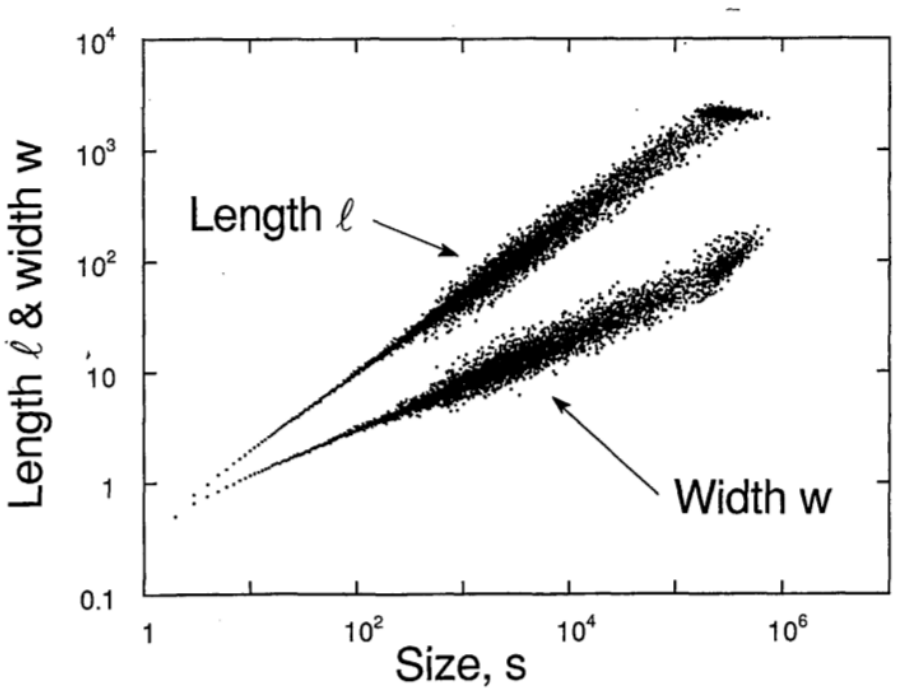
- $\beta/\nu_{||}$ as on figure
- z in a large sample
- Critical scaling of finite clusters



Directed percolation

- Density versus time

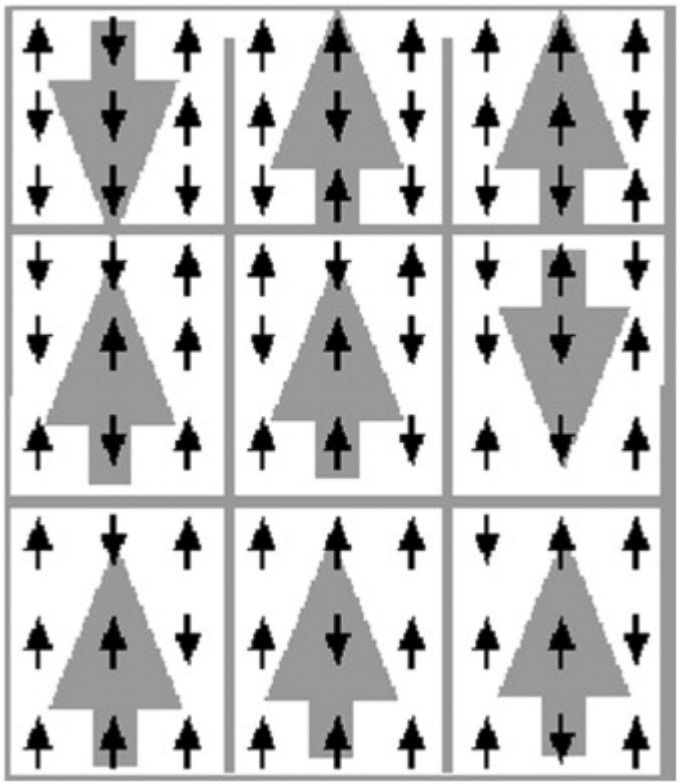


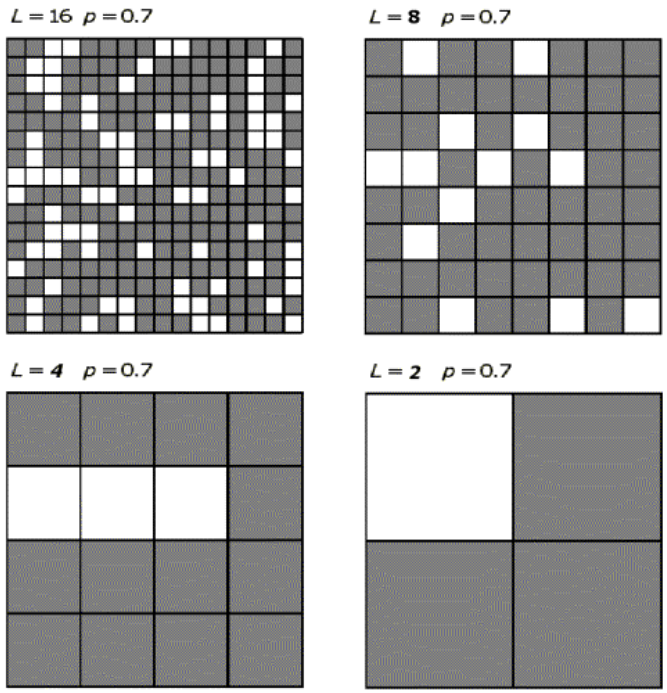


- Length/width versus size
- Clusters are fractal

Real space numerical renormalization group

- At the critical point the system is self similar (scale-free)
- It does not matter on which scale we are looking at it.

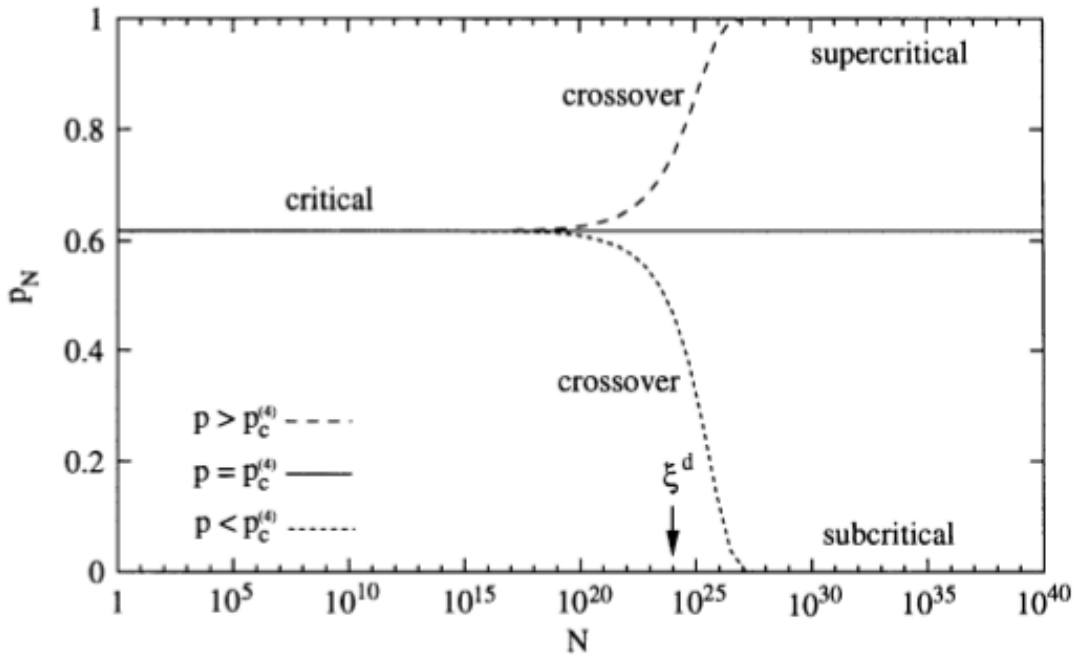




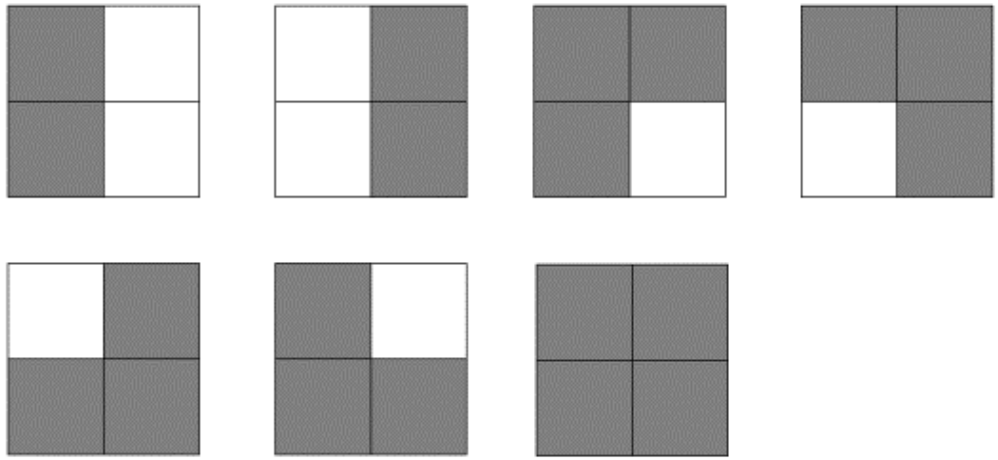
Real space numerical renormalization group

- As the system gets larger it converges into a fixed point

$$\lim_{n\to\infty} R_n(p) = \begin{cases} 0 & \text{for } 0 \leqslant p < p_c \,, \\ c & \text{for } p = p_c \,, \\ 1 & \text{for } p_c < p \leqslant 1 \end{cases}$$



Numerical renormalization group, percolation

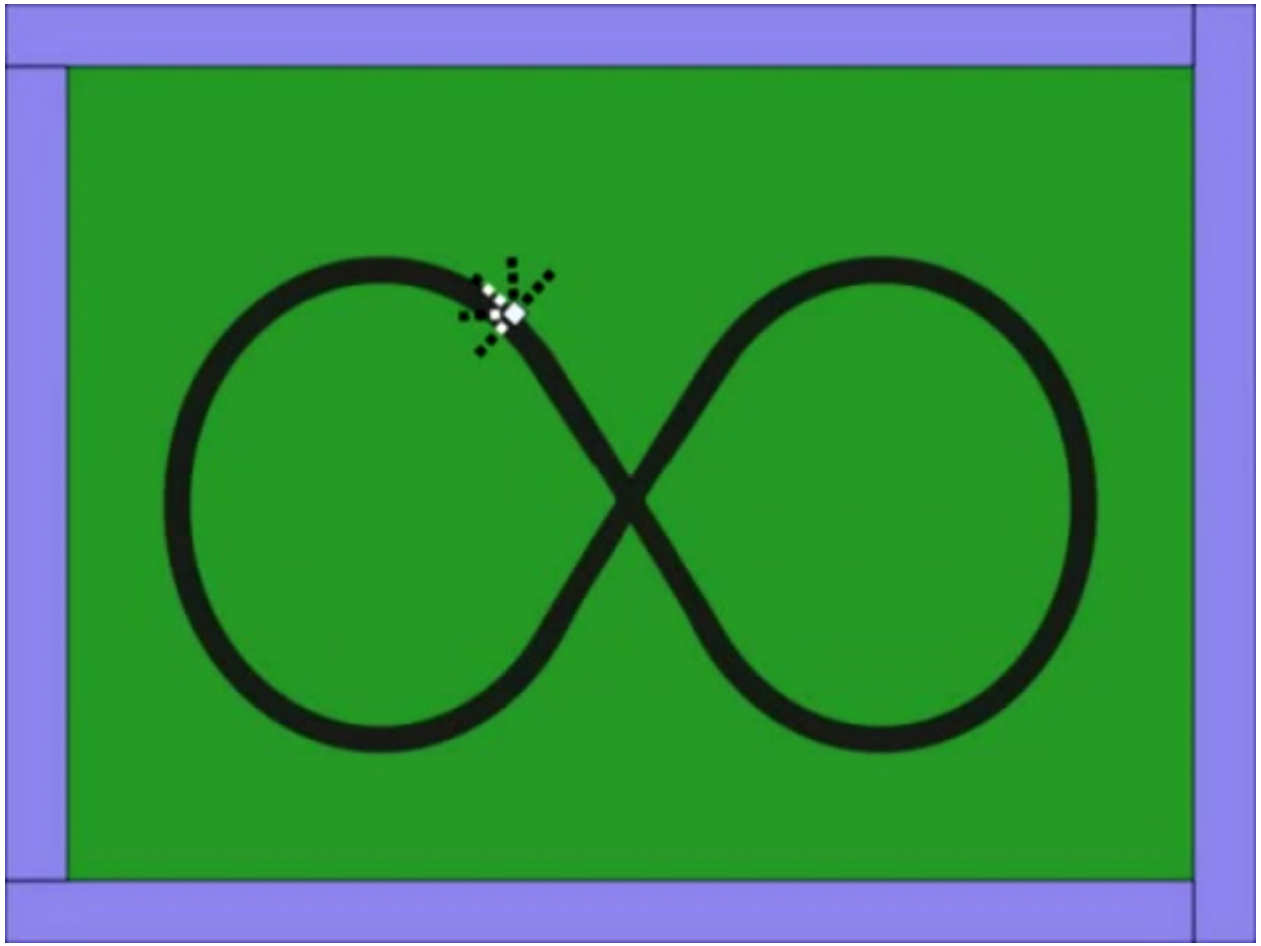


- probability that the cell is spanned:

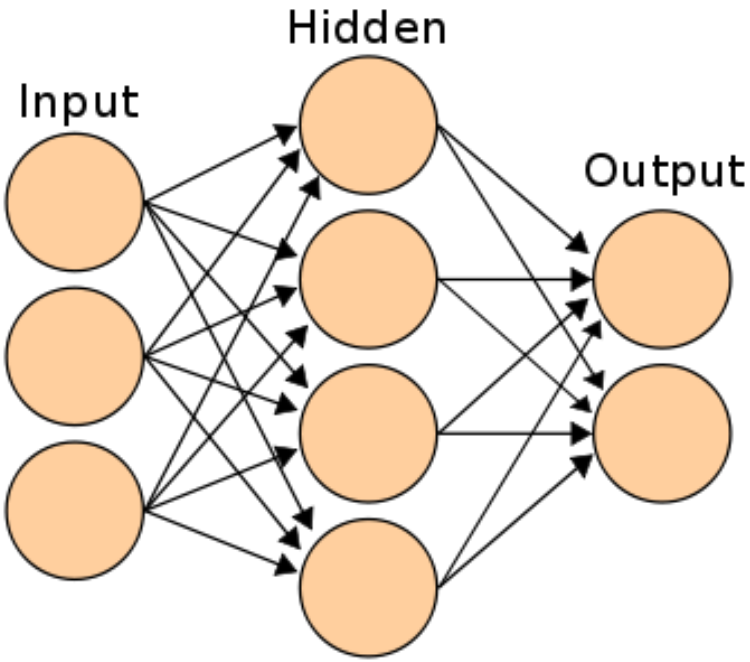
$$p' = R(p) = 2p^2(1-p)^2 + 4p^3(1-p) + p^4$$

- In the critical point $p' = p$.
- Three solutions $p_0 = 0$, $p_1 = 1$, and $p_* = 0.6180$
- Theoretical value $p_c = 0.5927$
- Larger blocks (only numerically possible) give better estimates

Neural networks

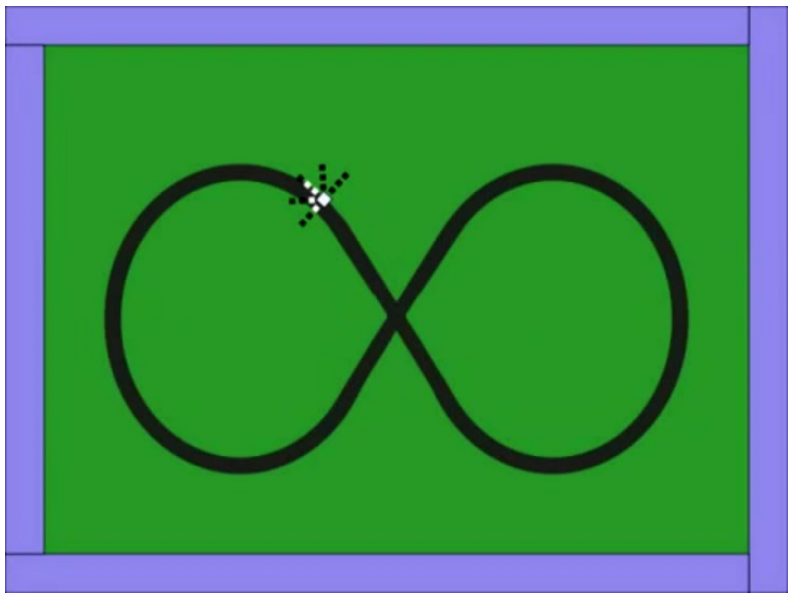


Neural networks



- Input pattern
- Output pattern
- Adaptive wights
- Approximating non-linear functions

- Machine learning
- Pattern recognition
- Handwriting
- Speech recognition



Neural networks

- Input vector I
- Output vector $O(I)$
- Transition matrix $W_{ij} \in [-1, 1]$
- Learning using a cost function
- Test goodness

Neural networks: Learning

- Supervised learning
- Data training:
  - Superwised learning
  - Fitness function, energy:

$$E = T(I) - O(I),$$

    where $T(I)$ is the target vector for input I
  - Minimize E for available set of $\{I, I(O)\}$ pairs
- Test goodness:
  - Use only part of $\{I, I(O)\}$ pairs for learning, the rest is for testing.
- Used for: pattern recognition, classification, etc.

Neural networks: Learning

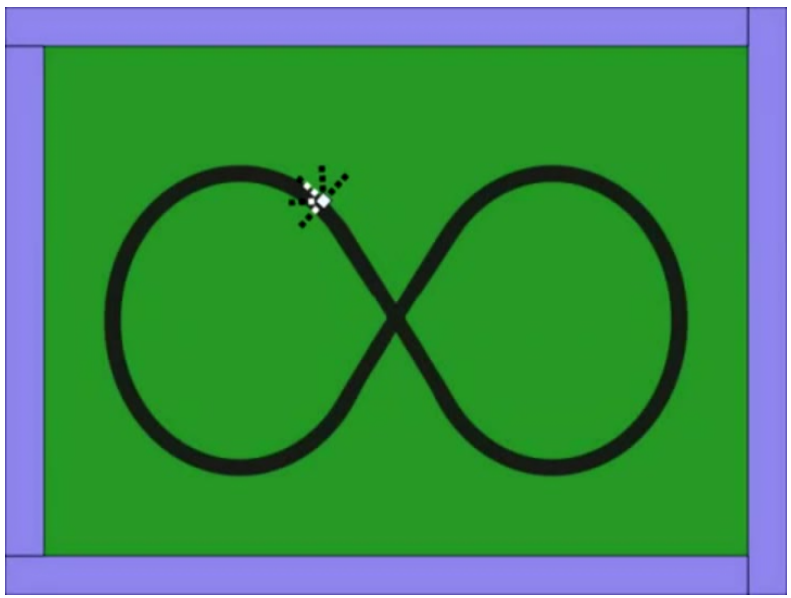
- Unsupervised learning
- Cost function may depend on task
- Cost function is deviation from mean data

$$C = E[(x - f(x))^2]$$

- Test goodness:
  - Some self consistent limit on the cost function
- Used for: estimation, filtering, etc.

Neural networks: Learning

- Reinforcement learning
- Cost function is a long time performance on an agent making decisions based on the neural network.
- Test goodness:
  - Compare with other agents which can be algorithmical or based on neural networks
- Used for: control problems, AI, complex optimization



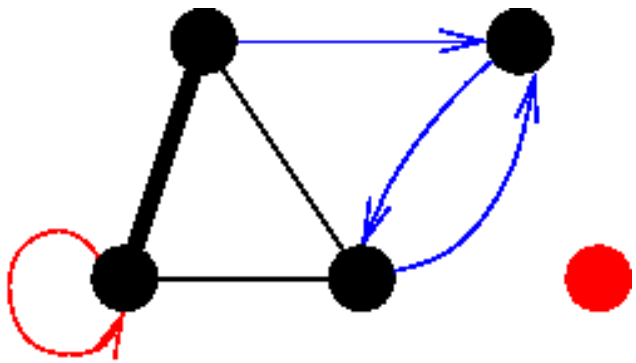
Neural networks

- Learning algorithms:
  - Linear regression
  - Genetic algorithm
  - Simulated annealing

Networks

Complex networks

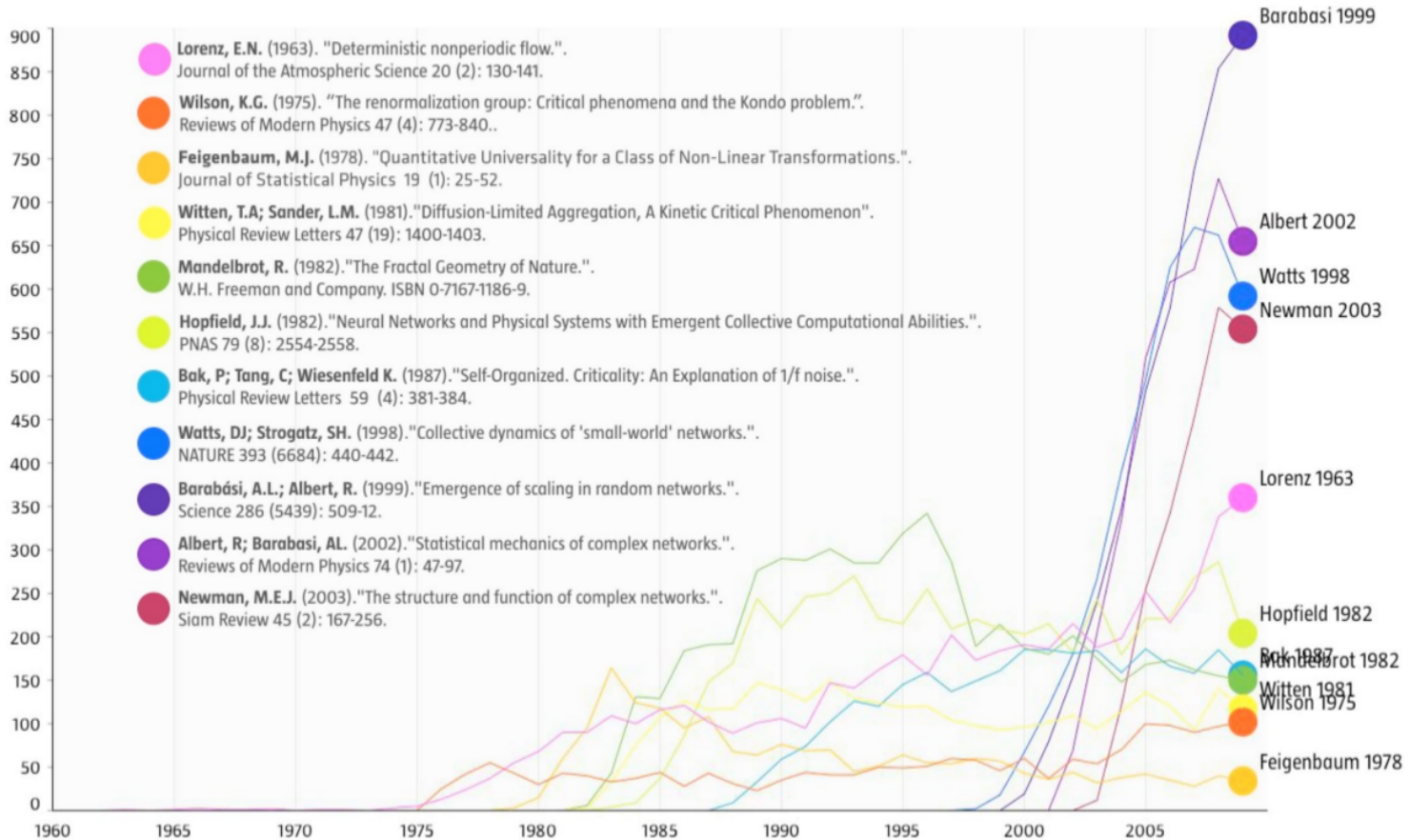
- Mathematics: Graphs
- Vertices, nodes, points
- Edges, links, arcs, lines
  - Directed or undirected
  - Loop
  - Multigraph
  - Wighted graphs
  - Connected



Complex networks

Phenomenon	Nodes	Links
Ising	Spins	Interaction(neighbors)
Cell metabolism	Molecules	Chem. reactions
Sci. collaboration	Scientists	Joint papers
WWW	Pages	URL links
Air traffic	Airports	Airline connections
Economy	Firms	Trading
Language	Words	Joint appearance

Complex networks, citations



Random Networks

Generate networks:

- From data:
  - Phone calls
  - WWW links
  - Biology database
  - Air traffic data
  - Trading data
- Generate randomly
  - From regular lattice by random algorithm (e.g. percolation)
  - Erdős-Rényi graph
  - Configurations model
  - Barabási-Albert model

Erdős-Rényi

- P. Erdős, A. Rényi, *On random graphs*, Publicationes Mathematicae Debrecen, Vol. 6 (1959), pp. 290-297 (cit 789)
- Two variants:
  1. $G(N, M)$: N nodes, M links
  2. $G(N, P)$: N nodes, links with p probability (all considered)
- Algorithm
  1. $G(N, M)$:
     - Choose i and j randomly $i, j \in [1, N]$ and $i \neq j$
     - If there is no link between i an j establish one
  2. $G(N, P)$: (Like percolation)
     - Take all $\{i, j\}$ pairs $(i \neq j)$
     - With probability p establish link between i and j

Erdős-Rényi

- Degree distribution

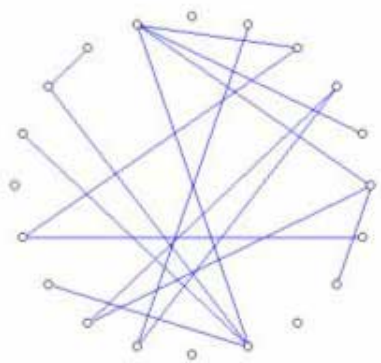
$$P(k) = \binom{N-1}{k} p^k (1-p)^{N-1-k}$$

- For large N and Np =const it is a Poisson distribution

$$P(k) \to \frac{(np)^k e^{-np}}{k!}$$

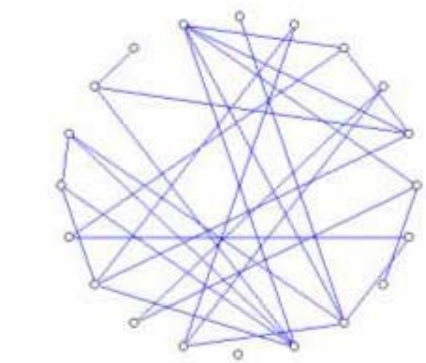
$p = 0$

(a)



$p = 0.1$

(b)

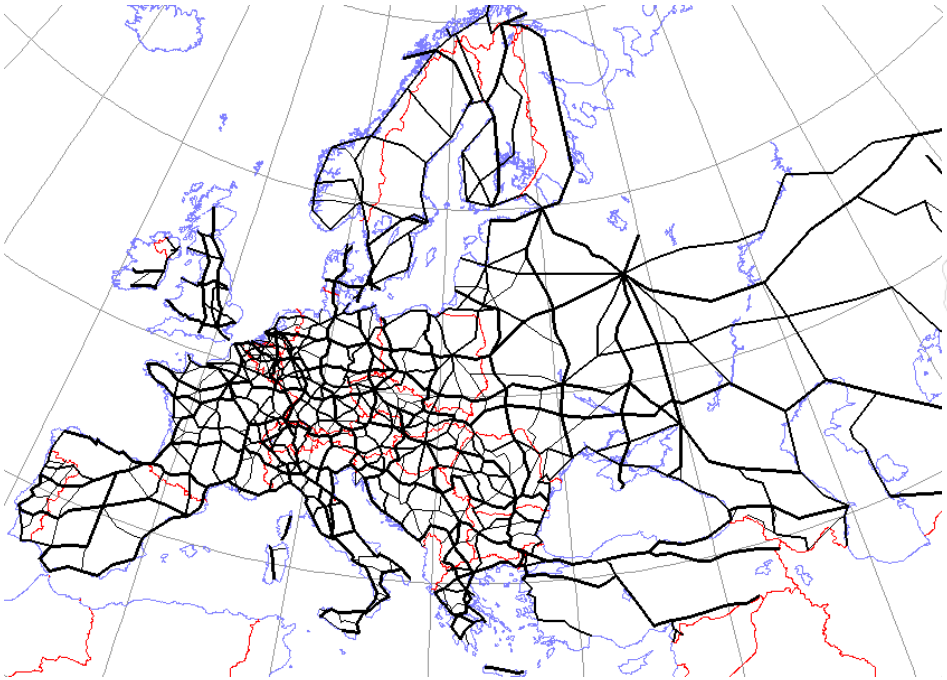


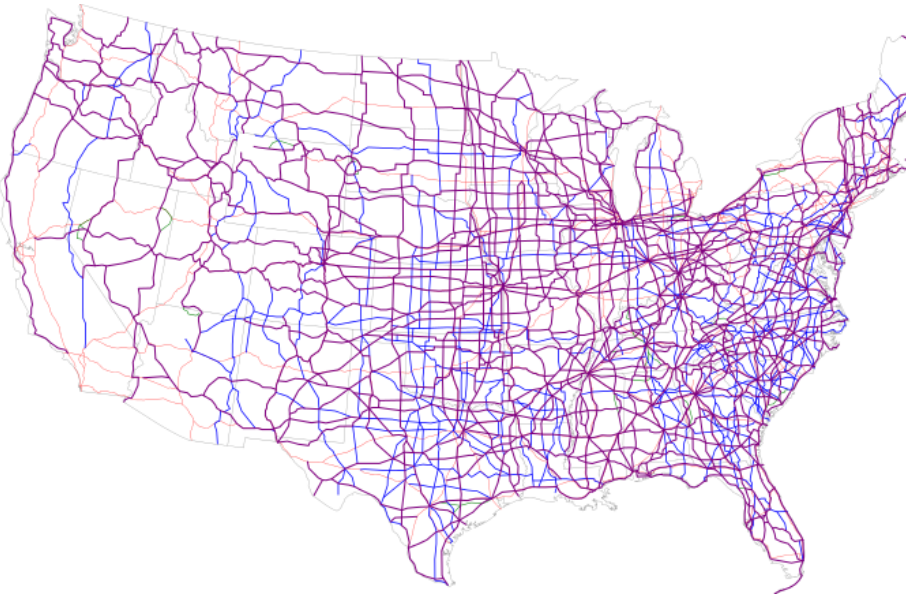
$p = 0.2$

(c)

Erdős-Rényi

- Real life: Read networks

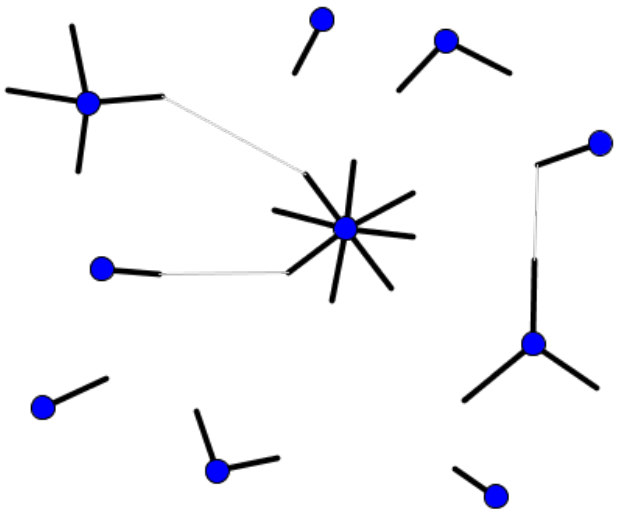


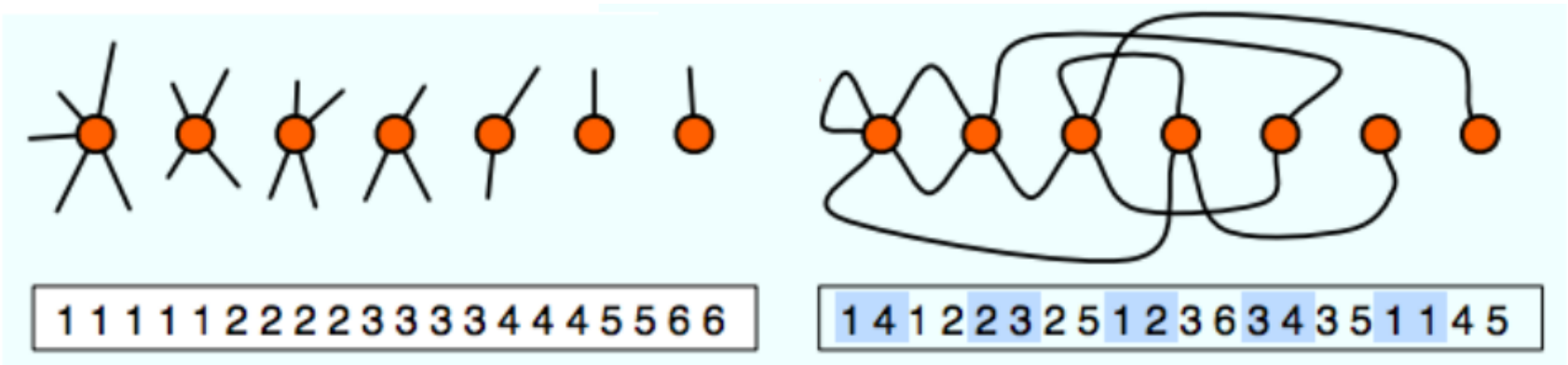


Most networks are different!

Configuration model

- Get the nodes ready with desired degree distribution
- Connect them randomly
- Self loops, and multiple links are created
- Problems at the end

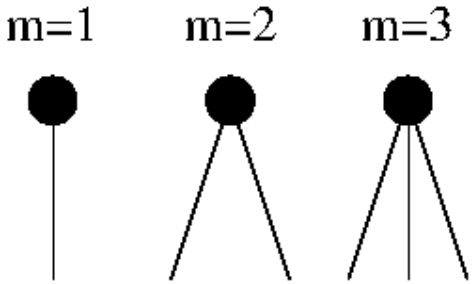




Preferential attachment

Barabási-Albert graph

- Initially a fully connected graph of m_0 nodes
- All new nodes come with m links ($m \leq m_0$)



- Links are attached to existing nodes with probability proportional to its number of links
- k_i is the number links of node i, then

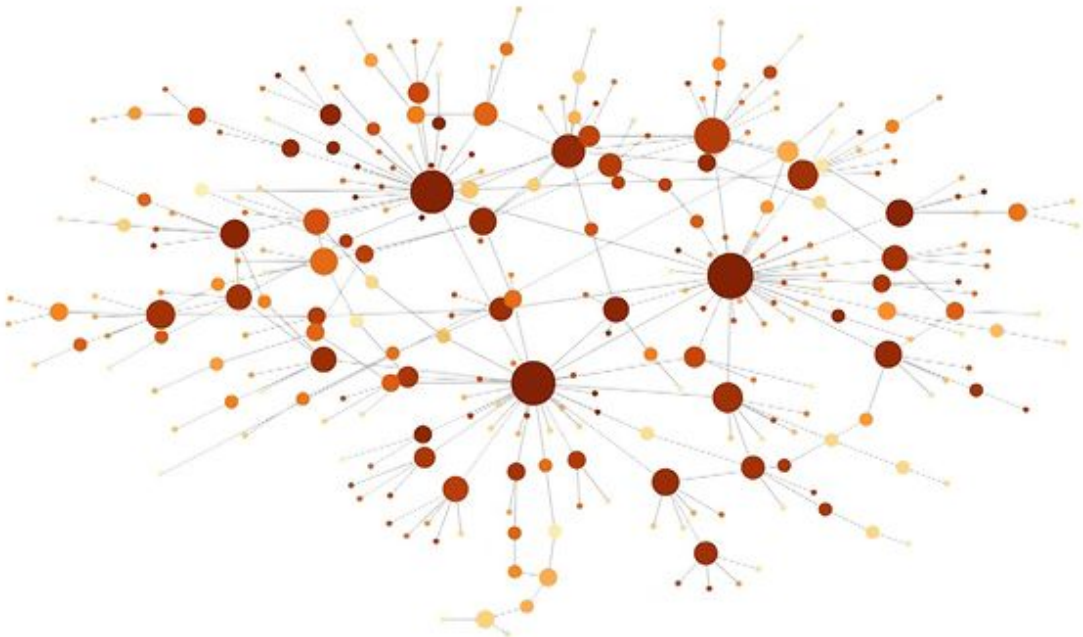
$$p_a = \frac{k_i}{\sum_j k_j}$$

Barabási-Albert graph

- Degree distribution

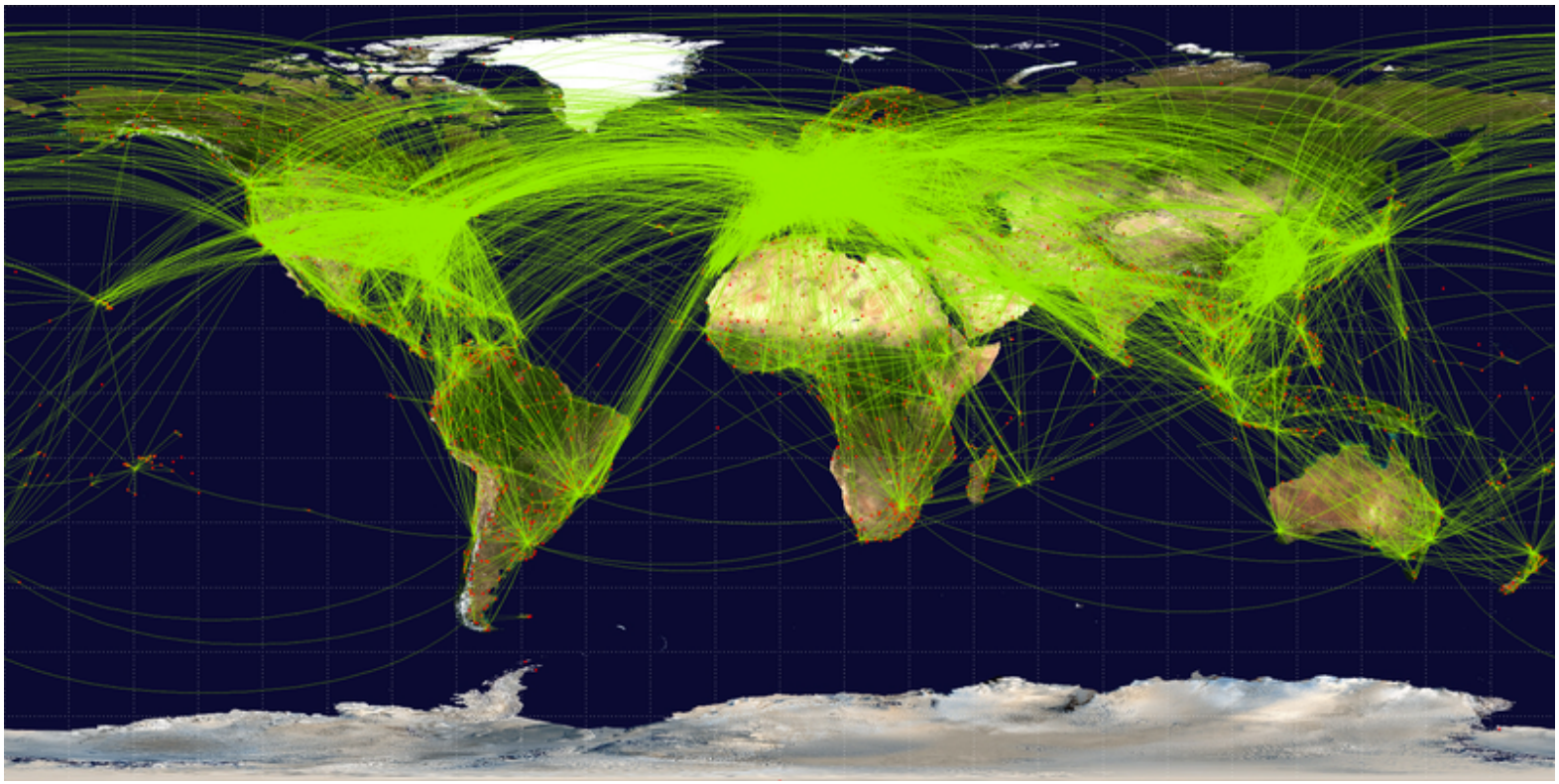
$$p(k) \sim k^{-3}$$

- Independent of m!

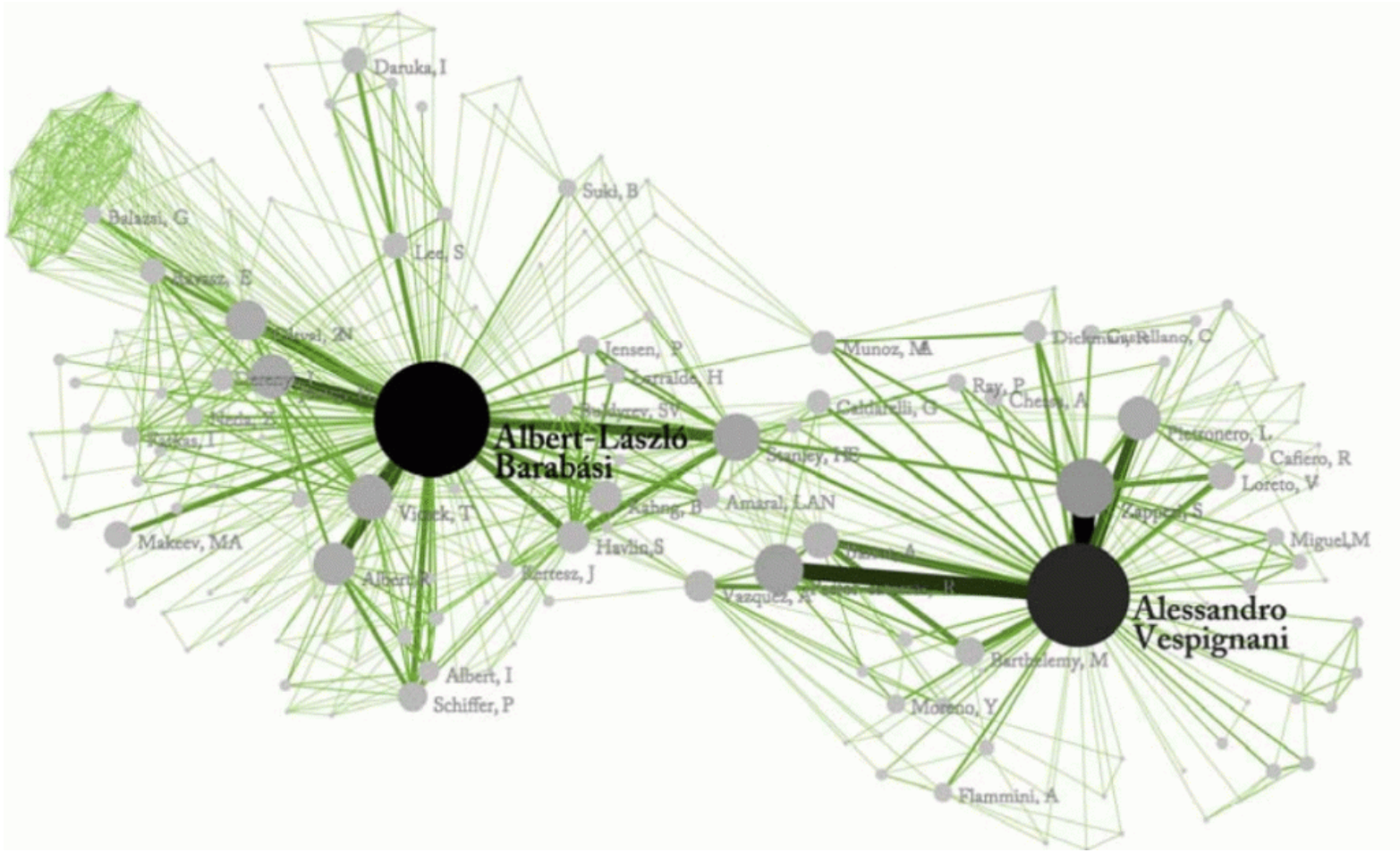


$m = 1$

Scalefree network example: Flight routes



Scalefree network example: Co-authorship

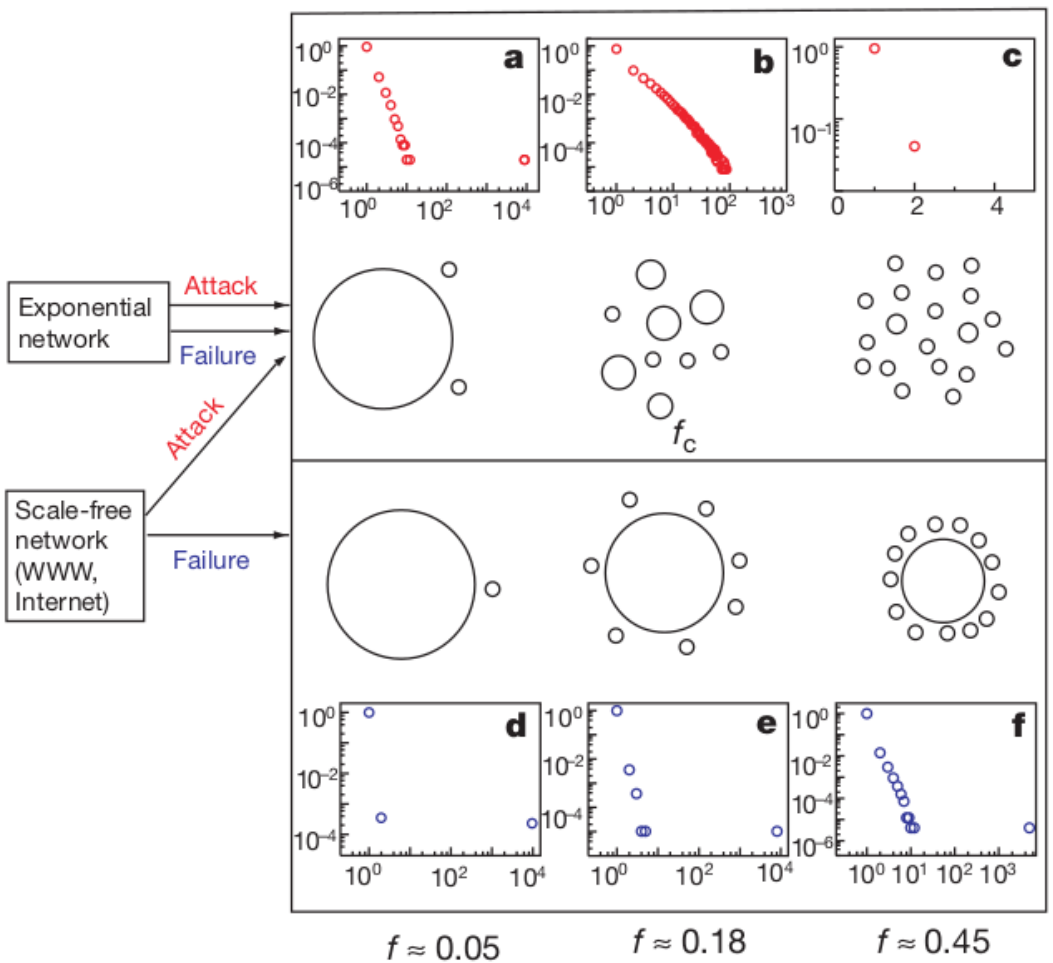


Algorithm for Barabási-Albert graph

1. $n = m_0$ number of existing nodes
2. $K = \sum_j k_j$ total number of connections
3. r random number $r \in [0, K]$
4. Find $i_{\max}$ for which $\sum_{j=0}^{i_{\max}} k_j < r$
5. If there is no edge then add one between nodes $n+1$ and $i_{\max}$
6. If node $n+1$ has less than m connections go to 3.
7. Increase n by 1
8. If $n < N$ go to 2.

Percolation and attack on random networks

- Failure: equivalent to percolation: remove nodes at random
- Attack: remove most connected nodes



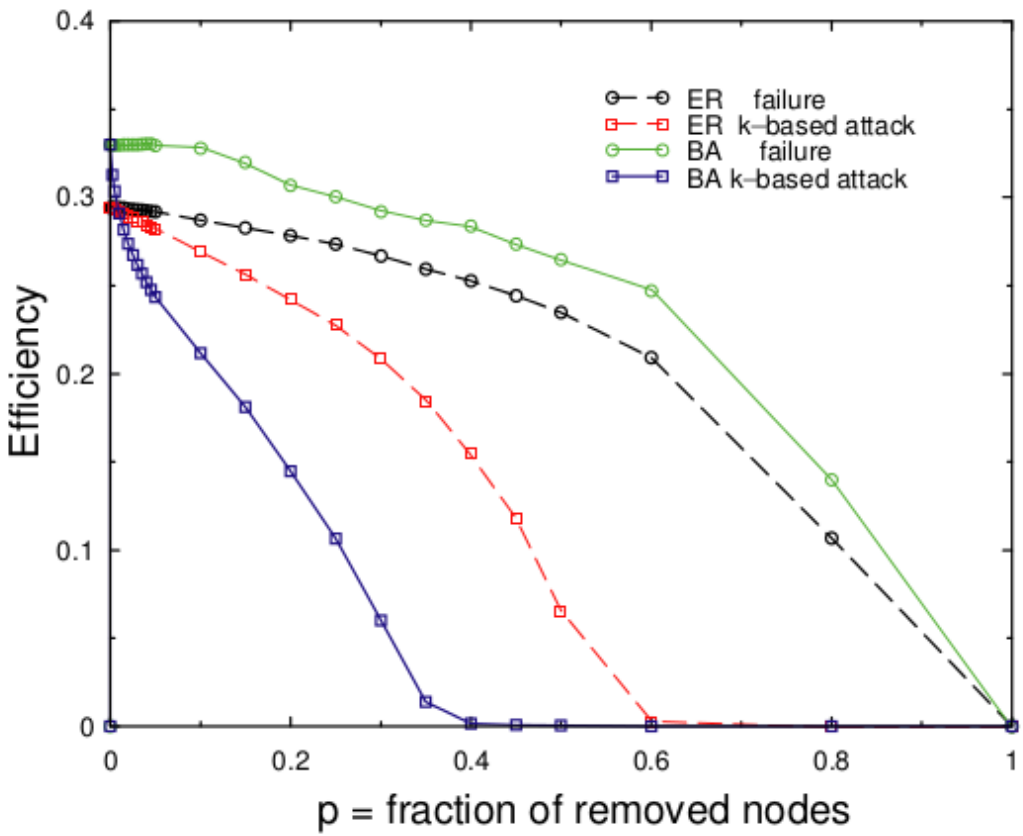
Percolation and attack on random networks

- Efficiency:

$$E(G) = \frac{1}{N(N-1)} \sum_{i \neq j} \frac{1}{t_{ij}}$$

t_{ij} the shortest path between i and j.

- $N = 2000$, $k = 10^4$



Percolation and attack on random networks