Simulations in Statistical Physics

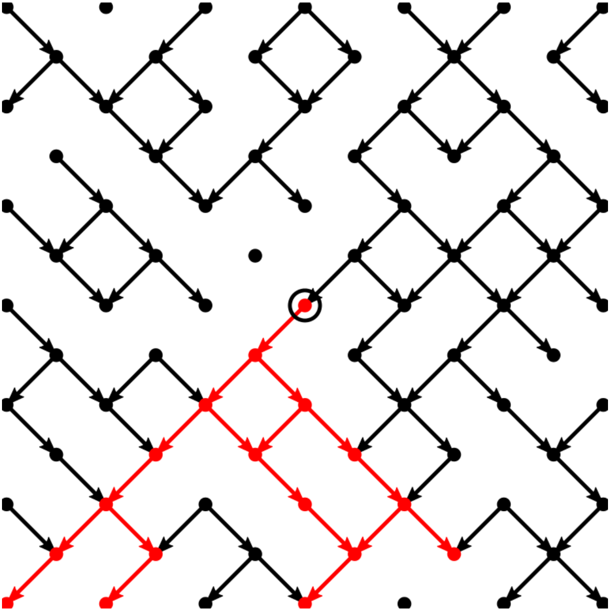
Course for MSc physics students

Janos Török

Department of Theoretical Physics

November 5, 2013

Directed percolation



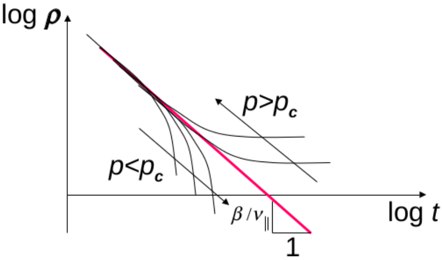
Directed percolation

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

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

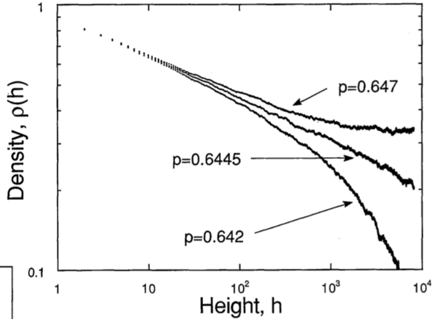
with $z = \nu_{||}/\nu_\perp$.

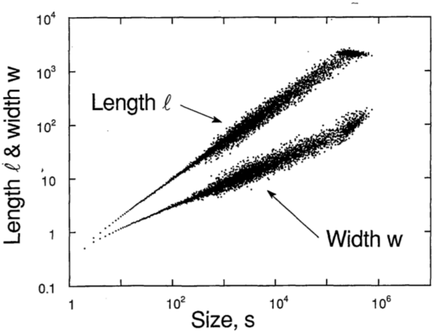
- $\beta/\nu_{||}$ as on figure
- z in a large sample
- Critical scaling of finite clusters



Directed percolation

- Density versus time

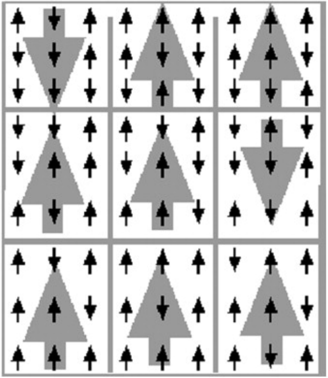


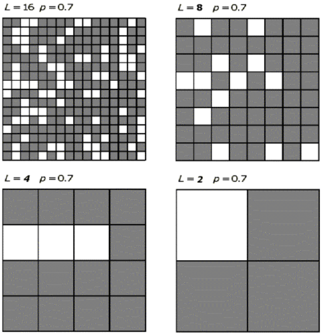


- Length/width versus size
- Clusters are fractal

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.

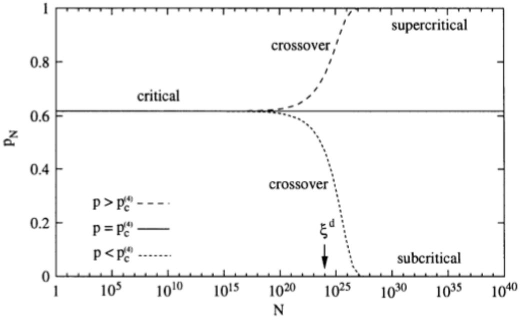




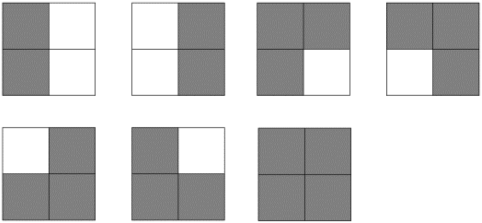
Numerical renormalization group

- 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

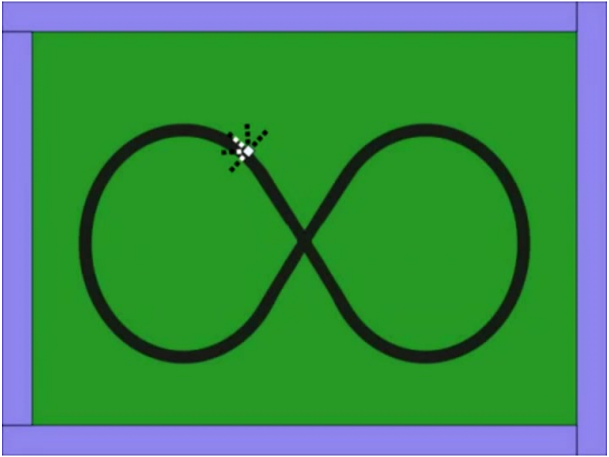


- probability that the cell is spanned:

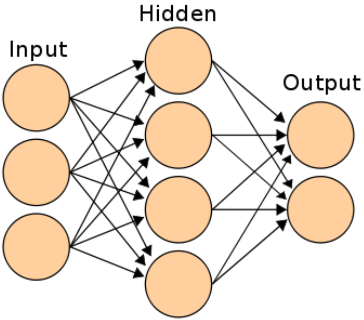
$$p' = R(p) = 2p^2(1-p)^2 + 4p^3(1-p) + p^4$$

- 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

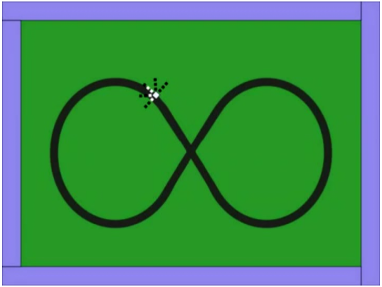


Neural networks



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

- Machine learning
- Pattern recognition
- Handwriting
- Speech recognition



Neural networks

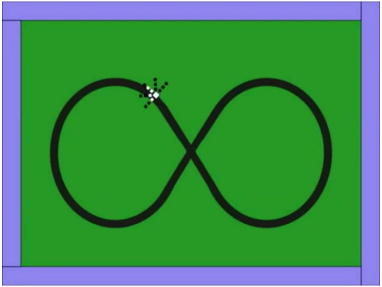
- Input vector I
- Output vector $O(I)$
- Transition matrix $W_{ij} \in [-1, 1]$
- Data training:
  - Superwised learning
  - Fitness function, energy:

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

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

Neural networks

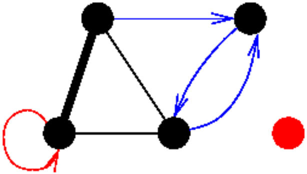
- Learning methods:
  - Linear regression
  - Genetic algorithm
  - Simulated annealing



Networks

Complex networks

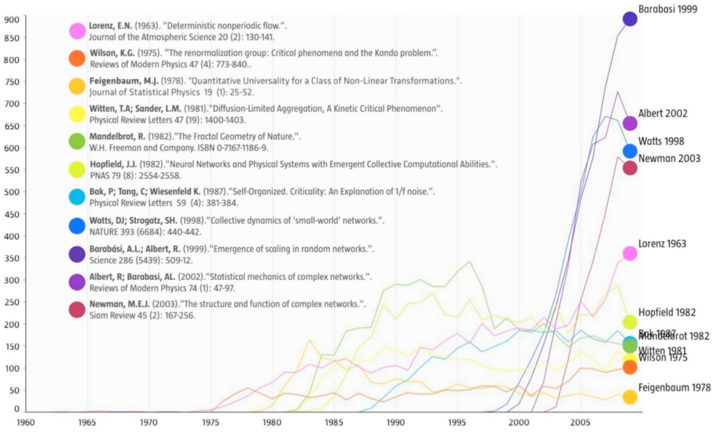
- Mathematics: Graphs
- Vertices, nodes, points
- Edges, links, arcs, lines
  - Directed or undirected
  - Loop
  - Multigraph
  - Wighted graphs
  - Connected



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

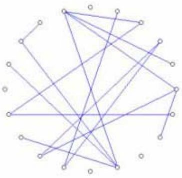
- Degree distribution

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

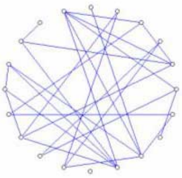
- For large N and Np =const it is a Poisson distribution

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

$p = 0$
(a)



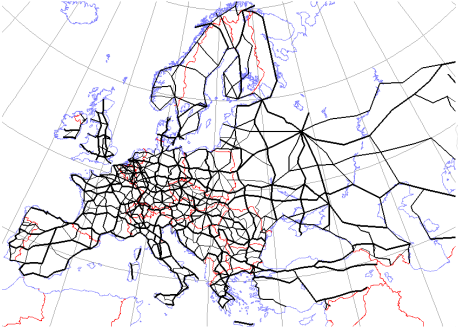
$p = 0.1$
(b)

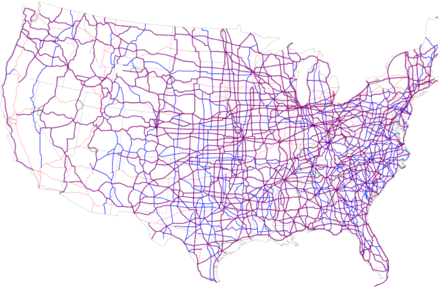


$p = 0.2$
(c)

Erdős-Rényi

- Real life: Read networks

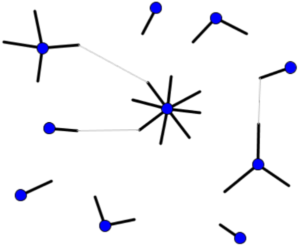


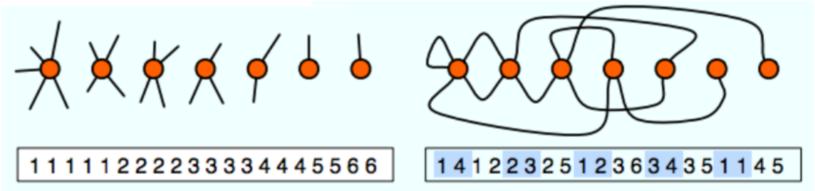


Most networks are different!

Configuration model

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

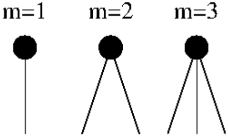




Preferential attachment

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

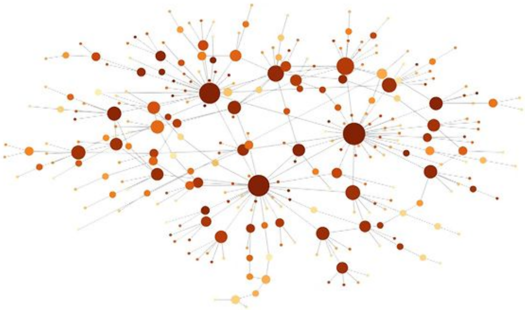
$$p_a = \frac{k_i}{\sum_j k_j}$$

Barabási-Albert graph

- Degree distribution

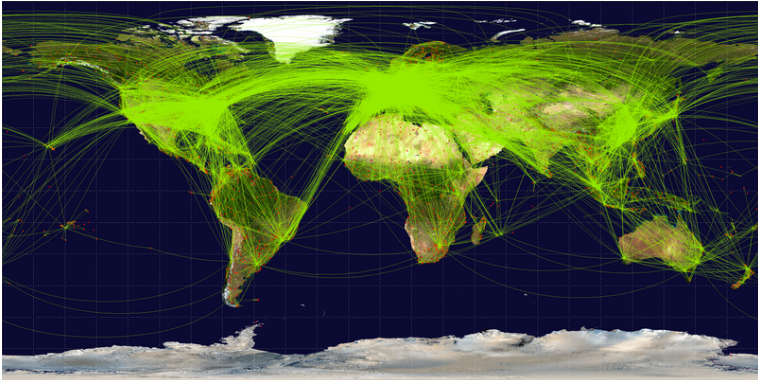
$$p(k) \sim k^{-3}$$

- Independent of m!

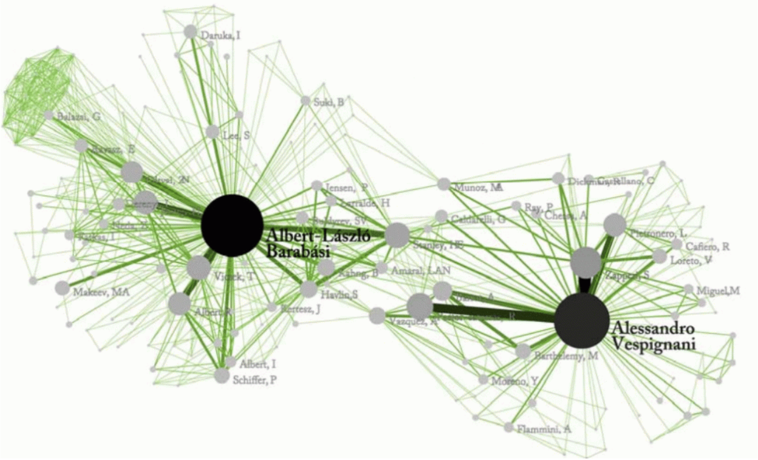


$m = 1$

Scalefree network example: Flight routes



Scalefree network example: Co-authorship

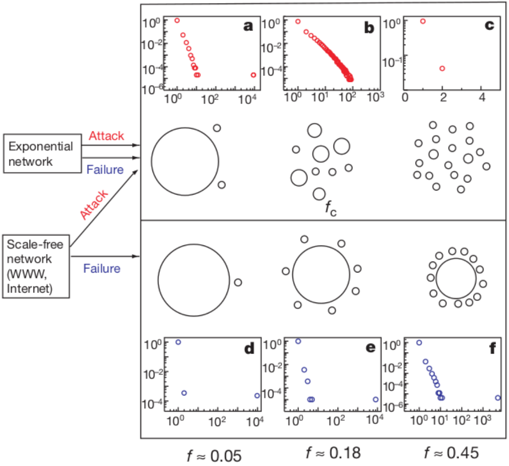


Algorithm for Barabási-Albert graph

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



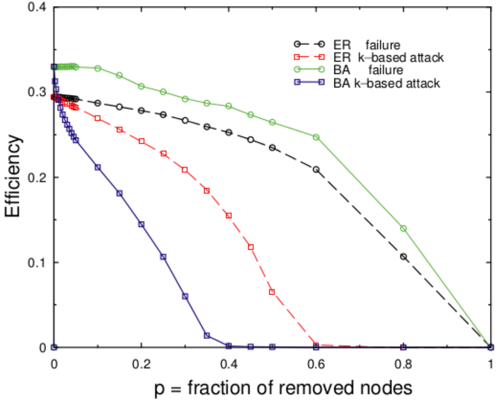
Percolation and attack on random networks

- 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