Neural Networks: Hopfield Model

(21 April 2003 — for four weeks)

Akira Imada

Brest State Technical University
e-mail: akira@bstu.by

last modified on:

June 3, 2003

□ Contents:

- What is Hopfield Model?
  - ⋆ A dynamical system with no input/output.
- Associative Memory.
  - ⋆ A model of human memory.
  - ⋆ How we model associative memory with Hopfield network?
- Combinatorial Optimization Problem.
  - ⋆ What is combinatorial optimization problem?
    - · NP-complete problem.
  - ⋆ Hopfield model can solve combinatorial optimization problem.
    - · Energy level of network state.
    - · How we solve the problems with Hopfield network?
    - · Examples.
      - 8-rook problem.
      - Traveling Salesperson Problem (TSP).
- Boltzman Machine.
  - ⋆ To escape from a trap of local minimum.

The 1st day:

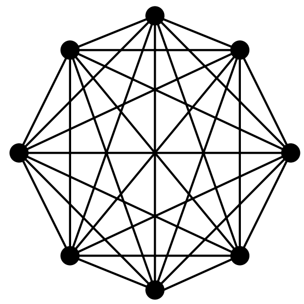
Monday, 21 April 2003
(16:20 – 17:40)

Today's Keywords:

neuron, synapse, weight, threshold, transfer function, dynamics of neurons' state, initial state, trajectory, limit cycle, chaotic trajectory, convergence to a stable state, fixed point

□ Hopfield Model

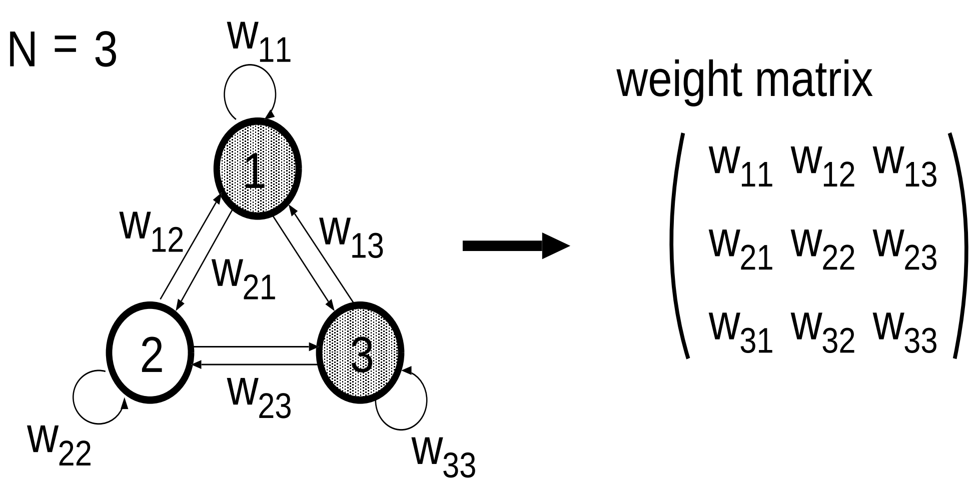
is a *Fully-connected Neural Network.*



□ State Transition

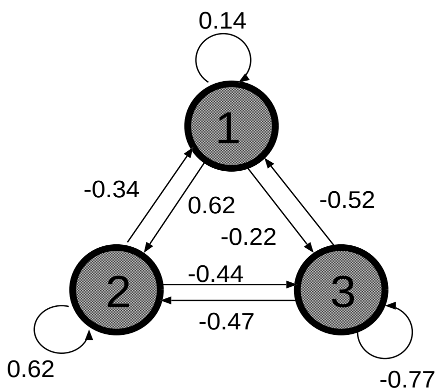
$$s_i(t+1) = sgn\left(\sum_{j \neq i}^{N} w_{ij} \cdot s_j(t)\right).$$

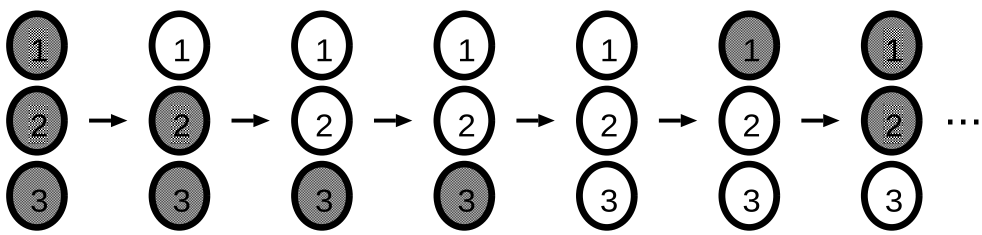
□ A Toy Model with Three Neurons



□ Examples of State Trajectory:

Example 1 (Chaotic Trajectry?)



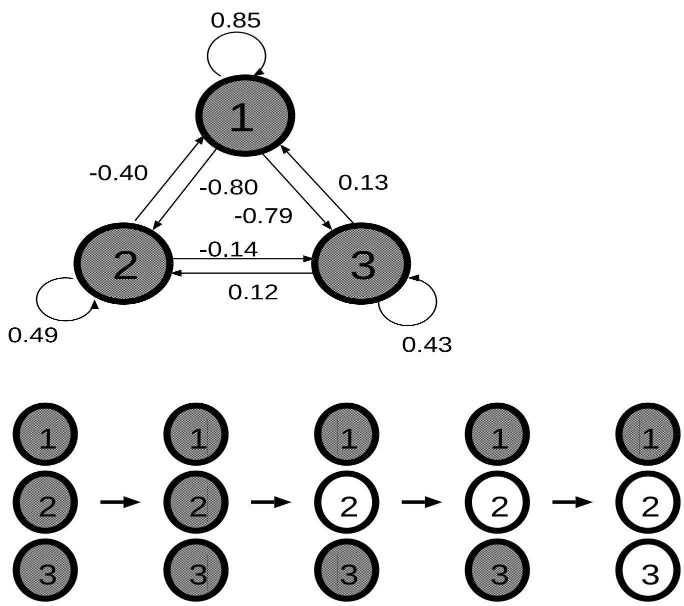


An example of state transition in the above toy example:

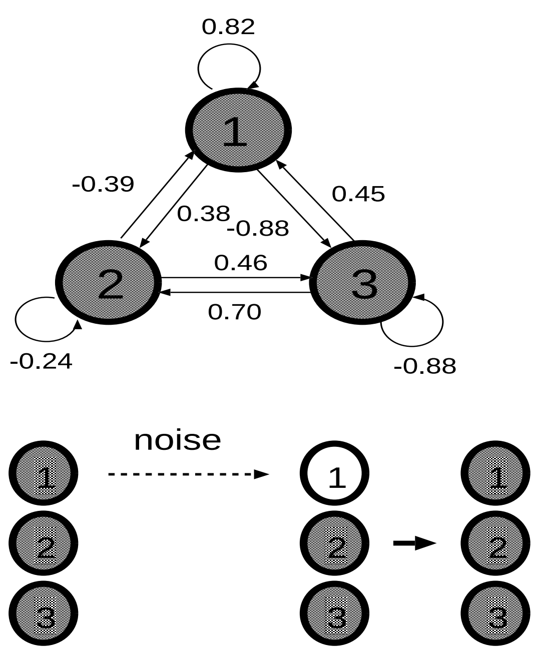
$$
\begin{aligned}
s_1(1) &= sgn\,(\,w_{11}\cdot s_1(0) + w_{12}\cdot s_2(0) + w_{13}\cdot s_3(0)\,)\\
&= sgn\,(\,0.14\cdot 1 + (-0.34)\cdot 1 + (-0.52)\cdot 1\,)\\
&= -1
\end{aligned}
$$

(cont'd)

Example 2: Convergence to a Stable State



Example 3: Fixed Point



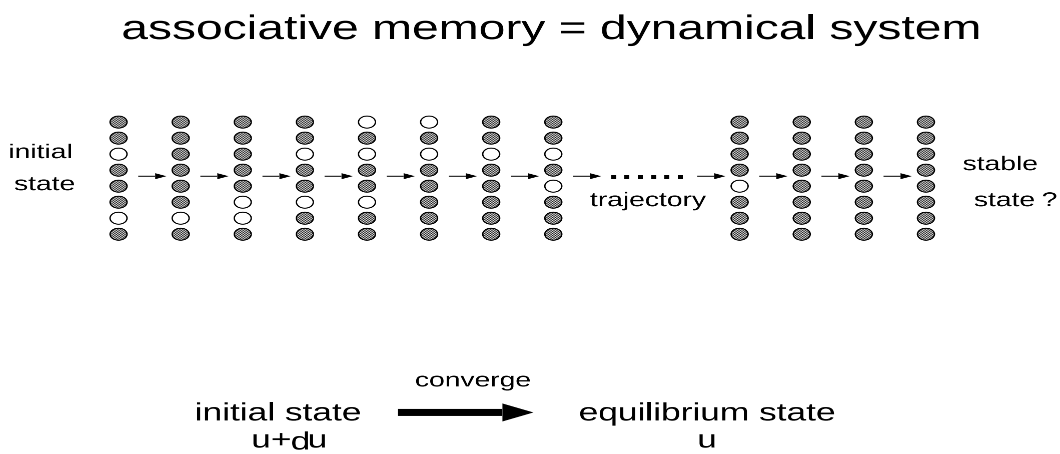
The 2nd day:

Monday, 28 April 2003
(16:20 – 17:40)

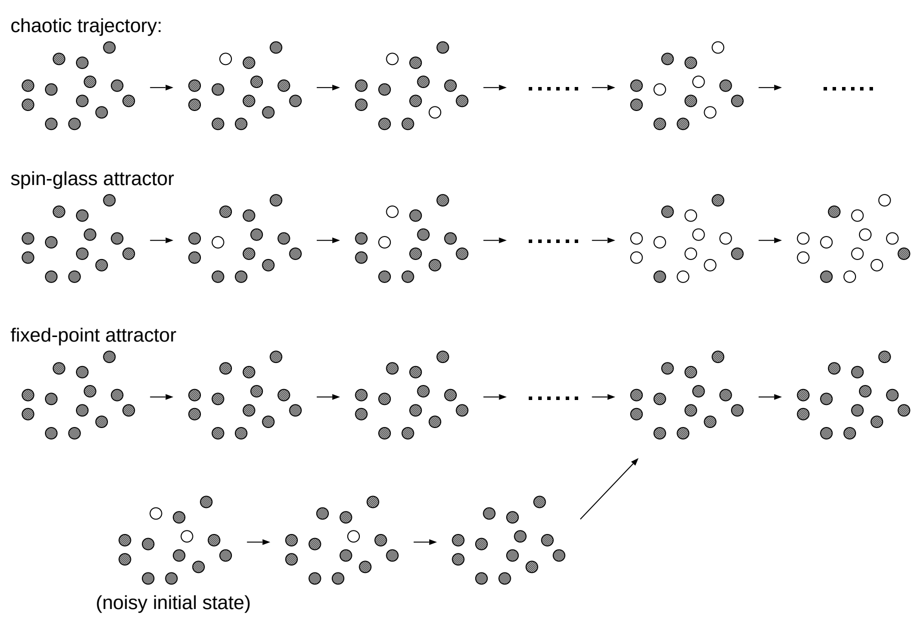
Today's Keywords:

associative memory, Hebbian learning
store/recall patterns,
storage capacity, basin of attraction

□ Associative Memory as a Dynamical System



□ A category of trajectories



□ Hebbian Weights

$$w_{ij} = \frac{1}{N} \sum_{\nu=1}^{p} \xi_i^\nu \xi_j^\nu \ (i \neq j), \ w_{ii} = 0.$$

- To store $(1, 1, 1)$, for example:

$$\begin{pmatrix} 0 & 0.11 & 0.11 \\ 0.11 & 0 & 0.11 \\ 0.11 & 0.11 & 0 \end{pmatrix}$$

- Many other solutions of a *weight configuration* to store a same set of patterns exist, and each of these solutions
  - ∗ can *recall* stored pattern from its
    - · partial input; and/or
    - · incorrect input
  - ∗ but has a different
    - · *basin of attraction*;
    - · *storage capacity*.

□ The Other Topics of Associative Memory

- Memory of a set of Motion Pictures

$$w_{ij} = \frac{1}{N}\sum_{\mu=1}^{p} \xi_i^{\mu+1}\xi_j^{\mu}. \tag{1}$$

- How to forget already stored patterns?

$$w_{ij} = w_{ij} - \frac{\lambda}{N}\xi_i\xi_j. \tag{2}$$

e.g., repeat equation(2) 1000 times with $\lambda = 0.01$.

- Storage capacity (How many patterns can be stored?)

· Under Hebb's rule

$$p/N < 0.138 \tag{3}$$

· E. Gardner asserts (1988).

$$p/N < 2 \tag{4}$$

The 3rd day:

Monday, 19 May 2003
(16:20 – 17:40)

Today's Keywords:

combinatorial optimization problem,
polynomial time, combinatorial explosion, NP-complete,
knapsack problem, weighted-matching-problem,
traveling salesperson problem,
only-one-bit-on-problem, eight-rook-problem,
virtual energy of Hopfield network for any possible state,
Garden of Eden

□ Combinatorial Optimization Problem.

When the problem is of size N [1]

$\Downarrow$

the number of possible candidate solutions is typically [2]

$$N!,\ N^N,\ e^N,\ \cdots$$

(called *combinatorial explosion*)
(Imagine how huge is the solution space for a large N.)

$\Downarrow$

If we want the one that minimizes the cost function

$\Downarrow$

we call these problems

Combinatorial Optimization Problems.

[1] In Traveling Salesperson Problem (TSP) the size is the number of city to which a salesperson must visit in a tour.
[2] In TSP the number of possible candidate solutions is $(N-1)!$.

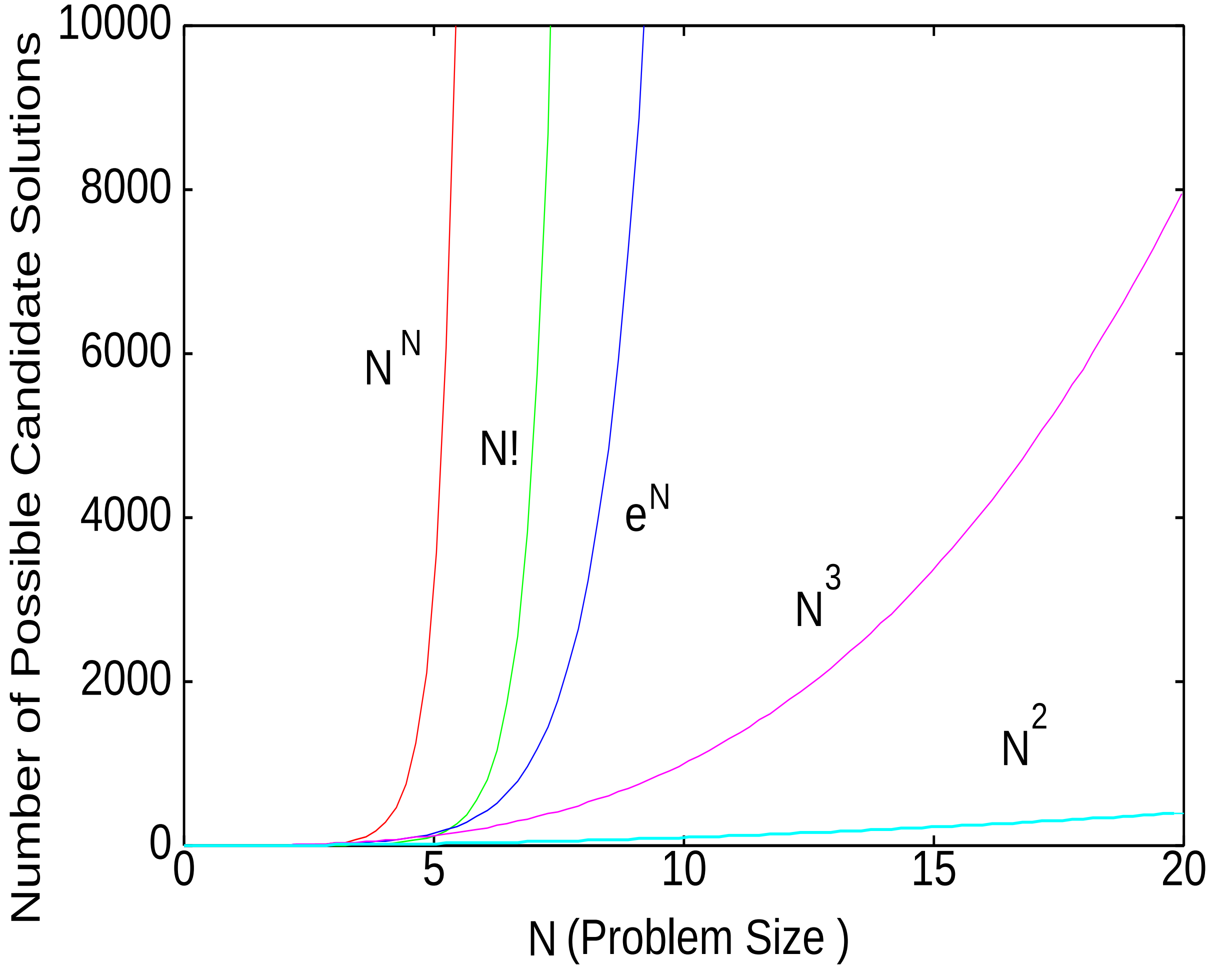
□ NP-Complete.

If there exists an algoithm that solves the problem in a time that grows only polynomially (or slower) w.r.t. N,

⇓

then it is said to be polynomial class: P.

- See examples of how the number grows w.r.t. N.



(cont'd)

If one can verify in polynomial time
whether any guess of the solution is right or not

$\Downarrow$

we call it non-deterministic polynomial class: *NP*.

If it's impossible to verify in polynomial time

$\Downarrow$

we call it *NP-hard*.

Then

$\Downarrow$

NP + *NP-hard* = *NP-complete* [3]

[3] If one could find a deterministic algorithm that solves one NP-complete problem in polynomial time then all other NP problems could be solved in polynomial time.

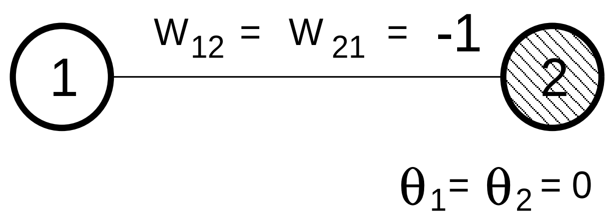
□ Energy of State

Hopfiled network in which synapse from neuron i to j is w_{ij} and threshold of neuron i is θ_i has the energy

$$E(\mathbf{x}) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}x_i x_j + \sum_{i=1}^{N}\theta_i x_i$$

when neurons' state is $\mathbf{x} = (x_1, x_2, \cdots, x_N)$.

- The simplest example of two neurons.



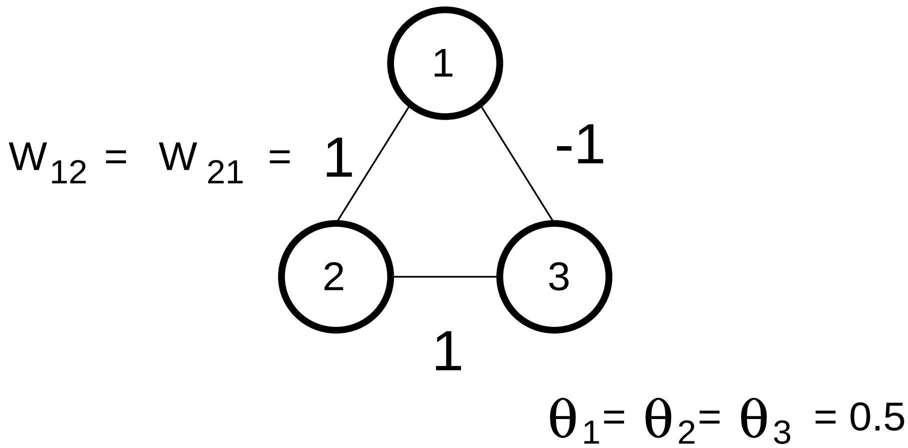
energey of the state $(-1, 1)$ and $(1, -1)$ is -1 while 1 for the state $(-1, -1)$ and $(1, 1)$, since

$$E(x_1, x_2) = -\frac{1}{2}(w_{12}x_1x_2 + w_{21}x_2x_1)$$

Exercise 1 *What will happen to this network?*

(cont'd)

- Yet another example: wight three neurons.



Exercise 2 *Calculate energy level for all the possible state of the network above.*

(cont'd)

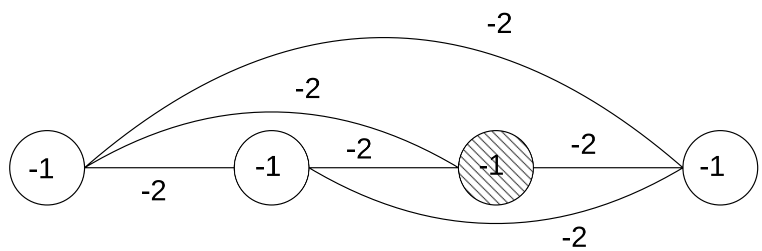
$$E(x_1, x_2, x_3) = -\frac{1}{2}(w_{12}x_1x_2 + w_{23}x_2x_3 + w_{31}x_3x_1) + (\theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3)$$

x_1	x_2	x_3	energy
–1	–1	–1	–2
–1	–1	+1	–1
–1	+1	–1	1
–1	+1	+1	0
+1	–1	–1	–1
+1	–1	+1	2
+1	+1	–1	0
+1	+1	+1	–1

energy	state of three neurons
2	(1 -1 1)
1	(-1 1 1)
0	(-1 1 1) (1 1 -1)
-1	(-1 -1 1) (1 -1 -1) (1 1 -1)
-2	(-1 -1 -1) ··· Garden of Eden

(cont'd)

- *Only-One-Bit-On Problem*



Only one x_i out of four should be one
with other three being zero.

$\Downarrow$

$(\sum_{i=1}^{4} x_i - 1)^2$ should be minimized.

$\Downarrow$

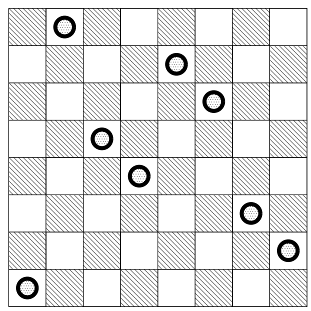
$$\begin{aligned}(\sum_{i=1}^{4} x_i - 1)^2 &= \sum_{i=1}^{4} x_i^2 + \sum_{i=1}^{4}\sum_{j=1}^{4} x_i x_j - 2\sum_{i=1}^{4} x_i + 1 \\ &= \sum_{i=1}^{4}\sum_{j=1}^{4} x_i x_j - \sum_{i=1}^{4} x_i + 1 \\ &= -\frac{1}{2}\sum_{i=1}^{4}\sum_{j=1}^{4}(-2)x_i x_j + \sum_{i=1}^{4}(-1)x_i + 1\end{aligned}$$

$\Downarrow$

$$\forall w_{ij} = -2 \;\&\; \forall \theta_i = -1$$

(cont'd)

- *Eight-Rook Problem*
  - ⋆ An extention of *Only-One-Bit-On Problem*
    - · Only one "1" in each raw and also only one "1" in each column.



the number of ones in column j is $\sum_{i=1}^{8} x_{ij}$

$\Downarrow$

As only single one in each column is allowed
$e1 = \sum_{j=1}^{8}(\sum_{i=1}^{8} x_{ij} - 1)^2)$ should be minimized.

$\Downarrow$

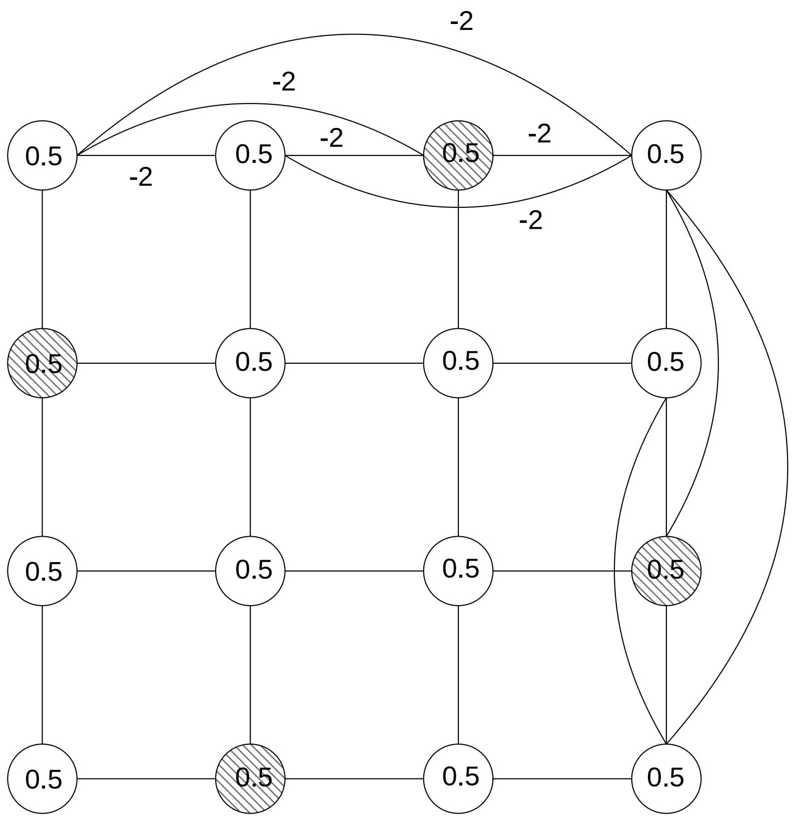
Also as only single one in each row is allowed
$e2 = \sum_{i=1}^{8}(\sum_{j=1}^{8} x_{ij} - 1)^2)$ should be minimzed too.

$\Downarrow$

Hence we define energy as $E(\mathbf{x}) = e1 + e2$

(cont'd)

- As can be easily guess, the network bellow solves the problem.



Exercise 3 *Implement the Hopfield Network above with $N = 8$ on your PC, and observe the trajectroy and final state starting with some initial state.*

The 4th day:

Monday, 26 May 2003
(16:20 – 17:40)

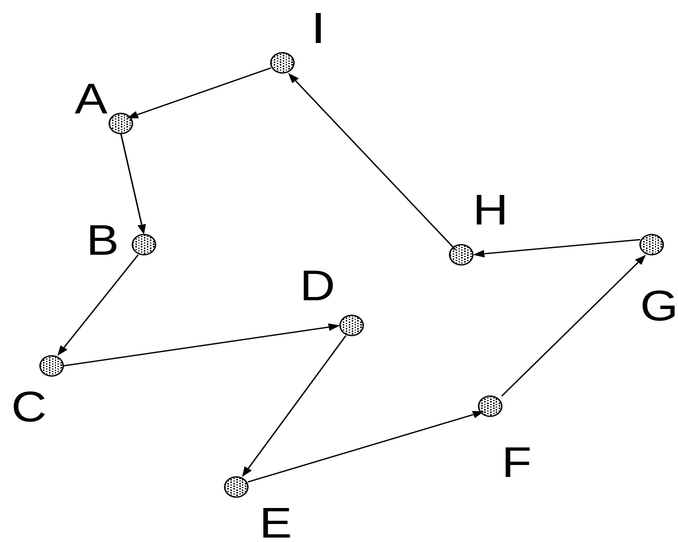
Today's Keywords:

traveling salesperson problem, Boltzman machine

□ Traveling Salesperson Problem (TSP)

A salesman should visit every city once but only once such that his traveling route is the shortest[4] .

An example:



tour:
A-B-C-D-E-F-G-H-I-A

Exercise 4 *Distribute* N *cities as points on your PC screen by defining all* $N(N-1)/2$ *distances between two cities.*

[4] With N cities, the number of all the possible route is

$$(N-1)! = \sqrt{2\pi N}\exp(N\ln N - N) \tag{5}$$

i.e., *not deterministic in polynomial time.*

(cont'd)

- Representation of a rout by a $N \times N$ matrix
  - ⋆ The rout 1-3-6-4-5-2 of 6-city TSP is represented as

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \qquad (6)$$

where "$i\alpha$-element is one" means that the i-th city is in the α-th location in the tour.

- When we denote distance between city i and j as d_{ij} then what should be mimimized will be

$$L = \frac{1}{2} \sum_{\alpha=1}^{N} \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{j\alpha} x_{j(\alpha+1)}$$

which implies iff both $x_{i\alpha}$ and $x_{j(\alpha+1)}$ are 1, d_{ij} is summed

(cont'd)

- So by defining energy as

$$\begin{aligned} E(\mathbf{x}) &= e1 + e2 + L \\ &= \sum_{j=1}^{8}(\sum_{i=1}^{8} x_{ij} - 1)^2) + \sum_{i=1}^{8}(\sum_{j=1}^{8} x_{ij} - 1)^2) \\ &\quad + \frac{1}{2}\sum_{\alpha=1}^{N}\sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}x_{j\alpha}x_{j(\alpha+1)} \end{aligned}$$

  and modifying this equation in the form of

$$E(\mathbf{x}) = -\frac{1}{2}\sum_{\imath=1}^{N}\sum_{\jmath=1}^{N} w_{ij}x_i x_j + \sum_{\imath=1}^{N} \theta_i x_i$$

  we obtain weight w_{ij} and threshold θ_i of the network

- Starting with a random configuration of the network state, it is expected to converge to the matrix corresponding to the shortest route.

□ Boltzman Machine

- A Hopfield network composed of N units in which

  ⋆ the state of neuron i is updated asynchronously according to $x_i = 1$ with probability p_i

  $$p_i = \frac{1}{1 + exp(-(\sum_{j=1}^{N} w_{ij}x_j - \theta_i)/T)}$$

  and $x_i = 0$ with probability $1 - p_i$, instead of *sgn* or *sigmoidal* transfer function.

  ⋆ To escape from an *local minimum*, increase *temperature* T.

  ⋆ An energy function defined on one-dimensional space (schematic).