Neural Networks
Lecture 6: Associative Memory II

H.A Talebi
Farzaneh Abdollahi

Department of Electrical Engineering

Amirkabir University of Technology

Winter 2011

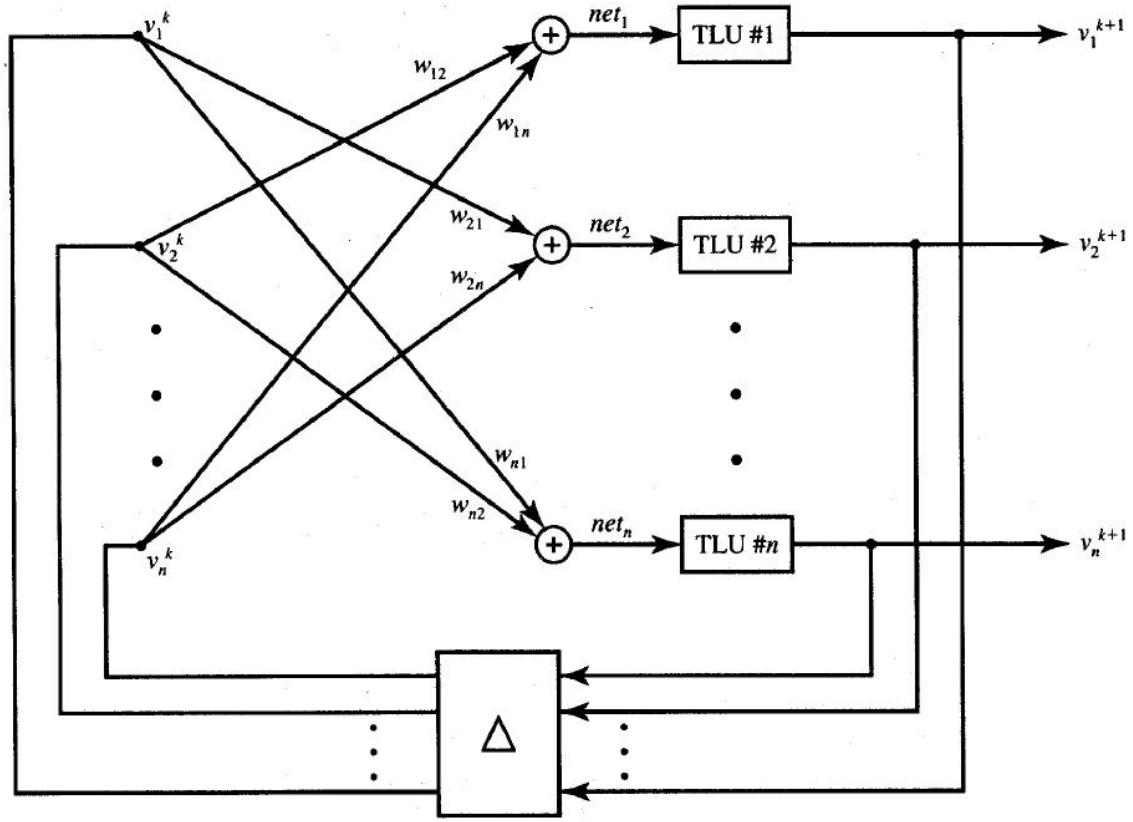
Hopfield Networks

- It is a special type of Dynamic Network that $v^0 = x^0$, i.e., $v^{k+1} = M[v^k]$
- It is a single layer feedback network which was first introduced by John Hopfield (1982,1988)
- Neurons are with either a hard-limiting activation function or with a continuous activation function
- In MLP:
  - The weights are updated gradually by teacher-enforced which was externally imposed rather than spontaneous
  - The FB interactions within the network ceased once the training had been completed.
  - After training, output is provided immediately after receiving input signal

- In FB networks:
  - The weights are usually adjusted spontaneously.
  - Typically, the learning of dynamical systems is accomplished without a teacher.
  - i.e., the adjustment of system parameters does not depend on the difference between the desired and actual output value of the system during the learning phase.
  - To recall information stored in the network, an input pattern is applied, and the network's output is initialized accordingly.
  - Next, the initializing pattern is removed and the initial output forces the new, updated input through feedback connections.
  - The first updated input forces the first updated output. This, in turn, produces the second updated input and the second updated response.
  - The transition process continues until no new updated responses are produced and the network has reached its equilibrium.
- $\therefore$ These networks should fulfill certain assumptions that make the class of networks stable and useful, and their behavior predictable in most cases.

- FB in the network
  - allows for great reduction of the complexity.
  - Deal with recalling noisy patterns
- Hopfield networks can provide
  - associations or classifications
  - optimization problem solution
  - restoration of patterns
  - In general, as with perceptron networks, they can be viewed as mapping networks
- One of the inherent drawbacks of dynamical systems is:
  - The solutions offered by the networks are hard to track back or to explain.

- w_{ij}: the weight value connecting the output of the jth neuron with the input of the ith neuron
- $W = \{w_{ij}\}$ is weight matrix
- $V = [v_1, ..., v_n]^T$ is output vector
- $net = [net_1, .., net_n]^T = Wv$
- $v_i^{k+1} = sgn(\sum_{j=1}^{n} w_{ij} v_j^k)$



- W is defined:

$$W = \begin{bmatrix} 0 & w_{12} & w_{13} & \dots & w_{1n} \\ w_{21} & 0 & w_{23} & \dots & w_{2n} \\ w_{31} & w_{32} & 0 & \dots & w_{3n} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ w_{n1} & w_{n2} & w_{n3} & \dots & 0 \end{bmatrix}$$

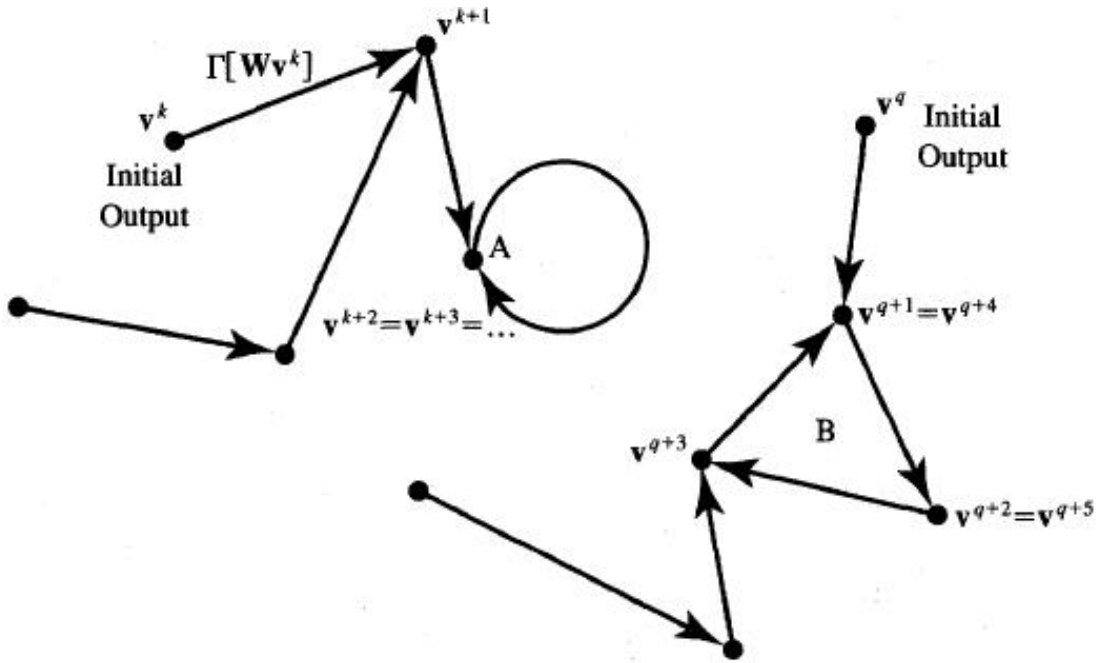
- It is assumed that W is symmetric, i.e., $w_{ij} = w_{ji}$
- $w_{ii} = 0$, i.e., There is no self-feedback
- The output is updated asynchronously. This means that
  - For a given time, only a single neuron (only one entry in vector V) is allowed to update its output
  - The next update in a series uses the already updated vector V.

- Example: In this example output vector is started with initial value V^0, they are updated by m, p and q respectively:

$$\begin{aligned} V^1 &= [v_1^0\ v_2^0\ \dots\ v_m^1\ v_p^0\ v_q^0\ \dots\ v_n^0]^T \\ V^2 &= [v_1^0\ v_2^0\ \dots\ v_m^1\ v_p^2\ v_q^0\ \dots\ v_n^0]^T \\ V^3 &= [v_1^0\ v_2^0\ \dots\ v_m^1\ v_p^2\ v_q^3\ \dots\ v_n^0]^T \end{aligned}$$

- The vector of neuron outputs V in n-dimensional space.
- The output vector is one of the vertices of the n-dimensional cube $[-1, 1]$ in E^n space.
- The vector moves during recursions from vertex to vertex, until it is stabilized in one of the 2^n vertices available.

- State transition map for a memory network is shown
- Each node of the graph is equivalent to a state and has one and only one edge leaving it.
- If the transitions terminate with a state mapping into itself, A, then the equilibrium A is fixed point.
- If the transitions end in a cycle of states, B, then we have a limit cycle solution with a certain period.
  - The period is defined as the length of the cycle. (3 in this example)



Stability Analysis

- To evaluate the stability property of the dynamical system of interest, let us study a so-called computational energy function.
- This is a function usually defined in n-dimensional output space V $E = -\frac{1}{2} V^T W V$
- In stability analysis, by defining the structure of W (symmetric and zero in diagonal terms) and updating method of output (asynchronously) the objective is to show that changing the computational energy in a time duration is nonpositive
- Note that since W has zeros in its diagonal terms, it is sign indefinite (its sign depend on the signs of v_j and v_i) $\rightsquigarrow E$ does not have minimum in unconstrained space
- However, the space v^n is bounded within $[-1, 1]$ hypercube, $\rightsquigarrow E$ has local minimum and showing E changes is nonpositive makes it a Lyapunov function

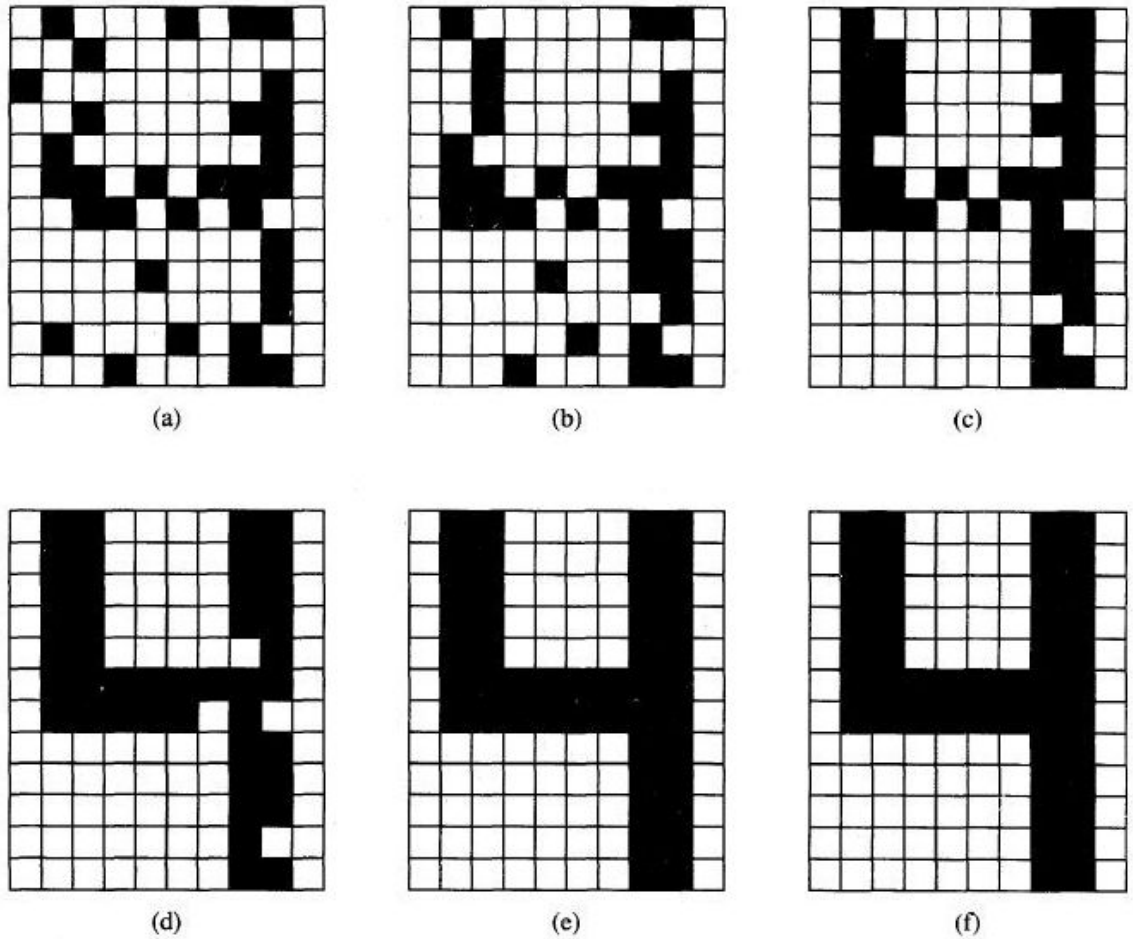
- The Energy increment is: $\triangle E = (\nabla E)' \triangle v$
- The energy gradient vector is :
  $\nabla E = -\frac{1}{2}(W' + W)v = -Wv = -net$ (W is symmetric)
- Assume in asynchronous update at the kth instant ith node of output is updated: $\triangle v = v^{k+1} - v^k = \begin{bmatrix} 0 \\ \vdots \\ \triangle v_i \\ \vdots \\ 0 \end{bmatrix}$
- $\therefore \triangle E = -net_i \triangle v_i$

- On the other hand output of the network v^{k+1} is defined based on output of TLU: $sgn(net_i) = \begin{cases} -1 & net_i < 0 \\ 1 & net_i \geq 0 \end{cases}$
- Hence
  - $net_i < 0 \rightsquigarrow \triangle v_i \leq 0$
  - $net_i > 0 \rightsquigarrow \triangle v_i \geq 0$
- $\therefore \triangle E$ is always neg.

- Energy function was defined as $E = -\frac{1}{2} v^T W v$
- In bipolar notation the complement of vector v is $-v$
- $\therefore E(-v) = -\frac{1}{2} v^T W v$
- $E(v) = E(-v) \rightsquigarrow \min E(v) = \min E(-v)$
- The memory transition may end up to v as easily as $-v$
- The similarity between initial output vector and v and $-v$ determines the convergence.
- It has been shown that synchronous state updating algorithm may yield persisting cycle states consisting of two complimentary patterns (Kamp and Hasler 1990)

Example

- A 10×12 bit map of black and white pixels representing the digit 4.
- The initial, distorted digit 4 with 20% of the pixels randomly reversed.



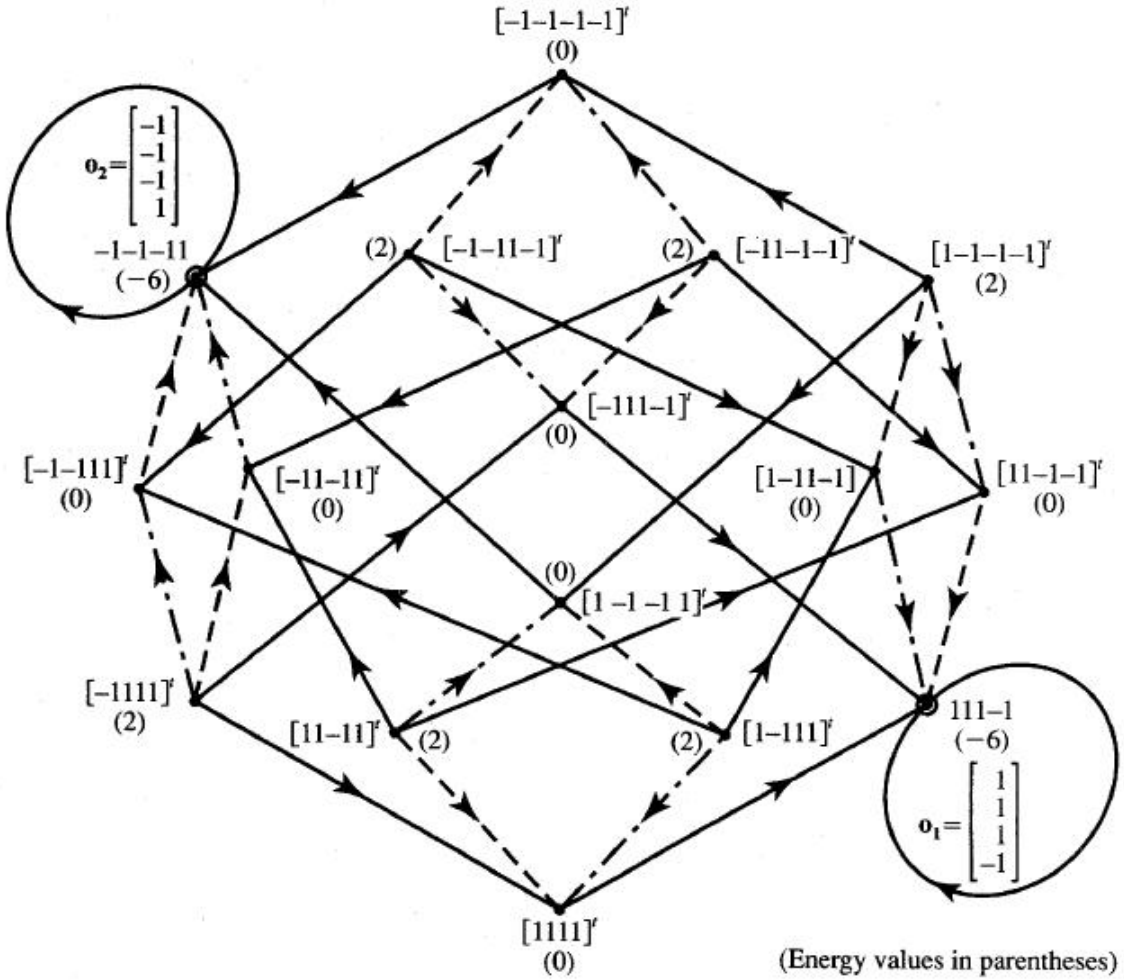
Example of recursive asynchronous update of corrupted digit 4: (a) $k = 0$,

- Example 2: Consider $W = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}, v^0 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$
- $v^1 = sgn(Wv) = sgn(\begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \end{bmatrix}) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$
- $v^2 = sgn(Wv) = sgn(\begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix}) = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$
- $v^0 = v^1 \rightsquigarrow$ It provides a cycle of two states rather than a fix point
- Example 3: Consider $W = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$
- The energy function becomes

$$E(v) = -\tfrac{1}{2}[v_1 \ v_2 \ v_3 \ v_4] \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} =$$

$$-v_1(v_2 + v_3 - v_4) - v_2(v_3 - v_4) + v_3 v_4$$

- It can be verifying that all possible energy levels are $-6,\ 0,\ 2$
- Each edge of the state diagram shows a single asynchronous state transition.
- Energy values are marked at cube vertexes
- By asynchronous updates, finally the energy ends up to its min value -6.



Energy levels and state transitions for the network

- Applying synchronous update:
  - Assume $v^0 = [1 \ -1 \ 1 \ 1]^T$
  - $v^1 = sgn(Wv^0) = [-1 \ 1 \ -1 \ -1]$
  - $v^2 = sgn(Wv^1) = [1 \ -1 \ 1 \ 1] = v^0$
  - It gets neither to lower energy (both are 2) nor fixed point
- **Storage Algorithm**
  - For bipolar prototype vectors: the weight is calculated: $W = \sum_{m=1}^{p} s^{(m)} s^{(m)T} - PI$ or $w_{ij} = (1 - \delta_{ij}) \sum_{m=1}^{p} s_i^{(m)} s_j^{(m)}$
  - δ_{ij} is Kronecker function: $\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$
  - If the prototype vectors are unipolar, the memory storage alg. is modified as $w_{ij} = (1 - \delta_{ij}) \sum_{m=1}^{p} (2s_i^{(m)} - 1)(2s_j^{(m)} - 1)$
  - The storage rule is invariant with respect to the sequence of storing pattern
  - Additional patterns can be added at any time by superposing new incremental weight matrices

- In $W = \sum_{m=1}^{p} s^{(m)} s^{(m)T}$ the diagonal terms are always positive and mostly dominant ($w_{ii} = \sum_{p=1}^{P} s_i^{(p)} s_i^{(p)T}$)
- Therefore, for updating the output, in Wv the elements of input vector v will be dominant and instead of the stored pattern
- $\therefore$ To avoid the possible problem the diagonal terms of W are set to zero.

Performance Analysis of Recurrent Autoassociative Memory

- Associative memories recall patterns that display a degree of "similarity" to the search argument.
- **Hamming Distance (HD)** is an index to measure this "similarity" precisely:
  - An integer equal to the number of bit positions differing between two binary vectors of the same length.

$$HD(x, y) = \frac{1}{2} \sum_{i=1}^{n} |x_i - y_i|$$

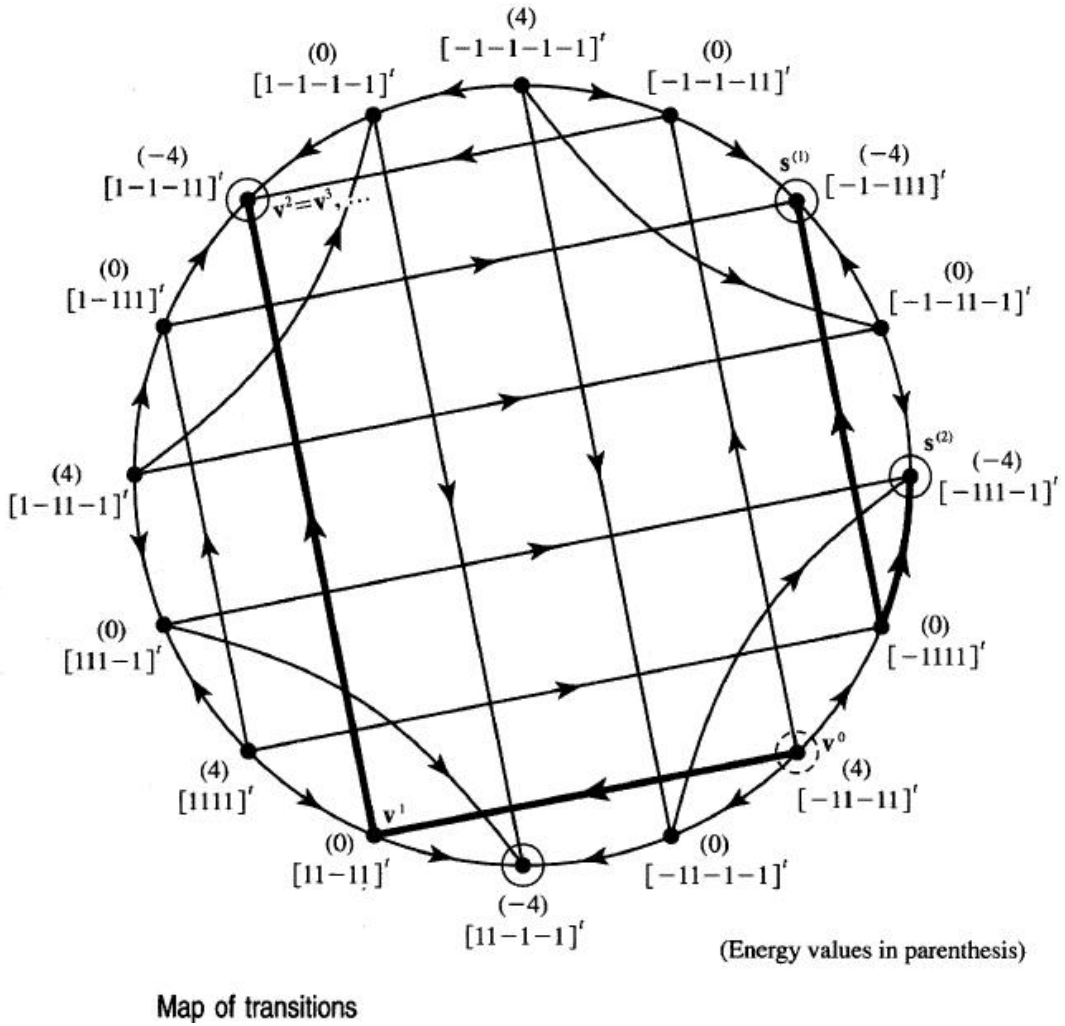
- $MAX\ HD = n$, it is between a vector and its component.
- The asynchronous update allows for updating of the output vector by $HD = 1$ at a time.

Example

- The patterns are
  $s^{(1)} = [-1 \ -1 \ 1 \ 1]$,
  $s^{(2)} = [-1 \ 1 \ 1 \ -1]$.
- The weight matrix is

$$\begin{bmatrix} 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \\ -2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \end{bmatrix}$$

- The energy function is
  $E(v) = 2(v_1 v_3 + v_2 v_4)$
- Assume $v^0 = [-1 \ 1 \ -1 \ 1]$, update asynchronously in ascending order



Map of transitions

$$\begin{aligned} v^1 &= [1\ 1\ -1\ 1]^T \\ v^2 &= [1\ -1\ -1\ 1]^T \\ v^3 &= v^4 = \ldots = [1\ -1\ -1\ 1]^T \end{aligned}$$

- It converges to negative image of $s^{(2)}$
- HD of v^0 is 2 from each patterns $\rightsquigarrow$ no particular similarity to any of them
- Now assume $v^0 = [-1\ 1\ 1\ 1]$: HD of v^0 to each of them is 1
- by ascending updating order:

$$\begin{aligned} v^1 &= [-1\ -1\ 1\ 1]^T \\ v^2 &= v^3 = \ldots = v^1 \end{aligned}$$

  - It reaches to $s^{(1)}$
- by descending updating order:

$$\begin{aligned} v^1 &= [-1\ 1\ 1\ -1]^T \\ v^2 &= v^3 = \ldots = v^1 \end{aligned}$$

- It has been shown that the energy is min along the following gradient vector direction $\nabla_v E(v) = -Wv = -(\sum_{p=1}^{P} s^{(p)} s^{(p)T} - PI)v = -\sum_{p=1}^{P} s^{(p)} s^{(p)T} v + Pv$
- Since scalar product of a and b is # of positions in which a and b are agree minus # of positions in which they are different: $a^T b = (n - HD(a,b)) - HD(a,b) = n - 2HD(a,b)$
- $\therefore \frac{\partial E}{\partial v_i} = -n \sum_{p=1}^{P} s_i^{(p)} + 2 \sum_{p=1}^{P} s_i^{(p)} HD(s^{(p)}, v) + Pv_i$
- When bit i of the output vector, v_i, is erroneous and equals - 1 and needs to be corrected to $+1$, the i'th component of the energy gradient vector must be negative.
- Any gradient component of the energy function is linearly dependent on HD , for $p = 1, 2, ..., P$.
- The larger the HD value, the more difficult it is to ascertain that the gradient component indeed remains negative due to the large potential contribution of the second sum term to the right side of expression

Performance Analysis

- Assume there are P patterns
- $v^0 = s^{(m)}$, where $s^{(m)}$ is one of the patterns

$$\begin{aligned} net &= Ws^{(m)} = \sum_{p=1}^{P} (s^{(p)} s^{(p)T} - PI) s^{(m)} \\ &= ns^{(m)} - Ps^{(m)} + \eta \\ \eta &= \sum_{p \neq m}^{P} s^{(p)} s^{(p)T} s^{(m)} \end{aligned}$$

- Hint: Assume all elements of patterns are bipolar $\rightsquigarrow s^{(m)T} s^{(m)} = n$

1. If the stored patterns are orthogonal: $(s^{(p)T}s^{(m)} = 0; s^{(m)T}s^{(m)} = n)$

$$net = Ws^{(m)} = (n - P)s^{(m)}$$

- $\therefore s^{(m)}$ is an eigen vector of W and an equilibrium of network

$$E(v) = -\frac{1}{2}v^T(\sum_{p=1}^{P}(s^{(p)}s^{(p)T})v + \frac{1}{2}v^T PIv$$

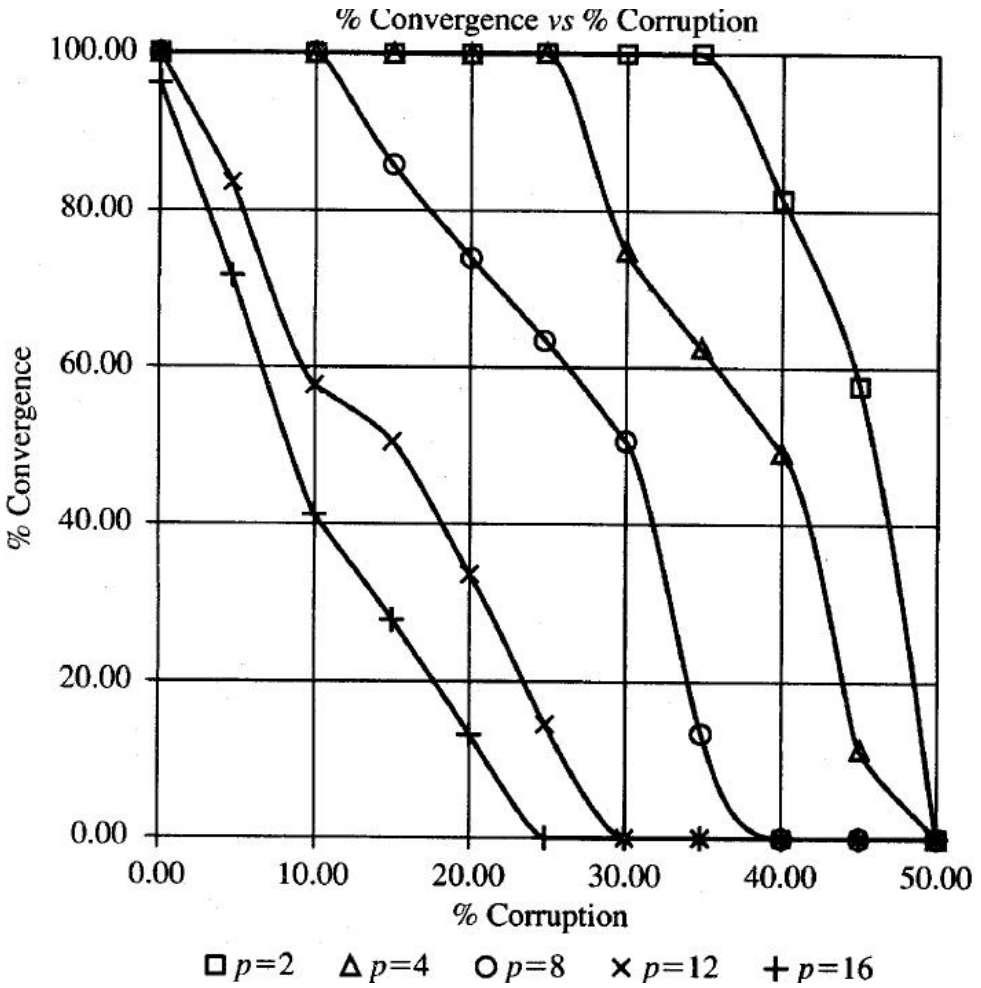
$$E(s^{(m)}) = -\frac{1}{2}(n^2 - Pn)$$

- If $n > P$, the energy at this pattern is min

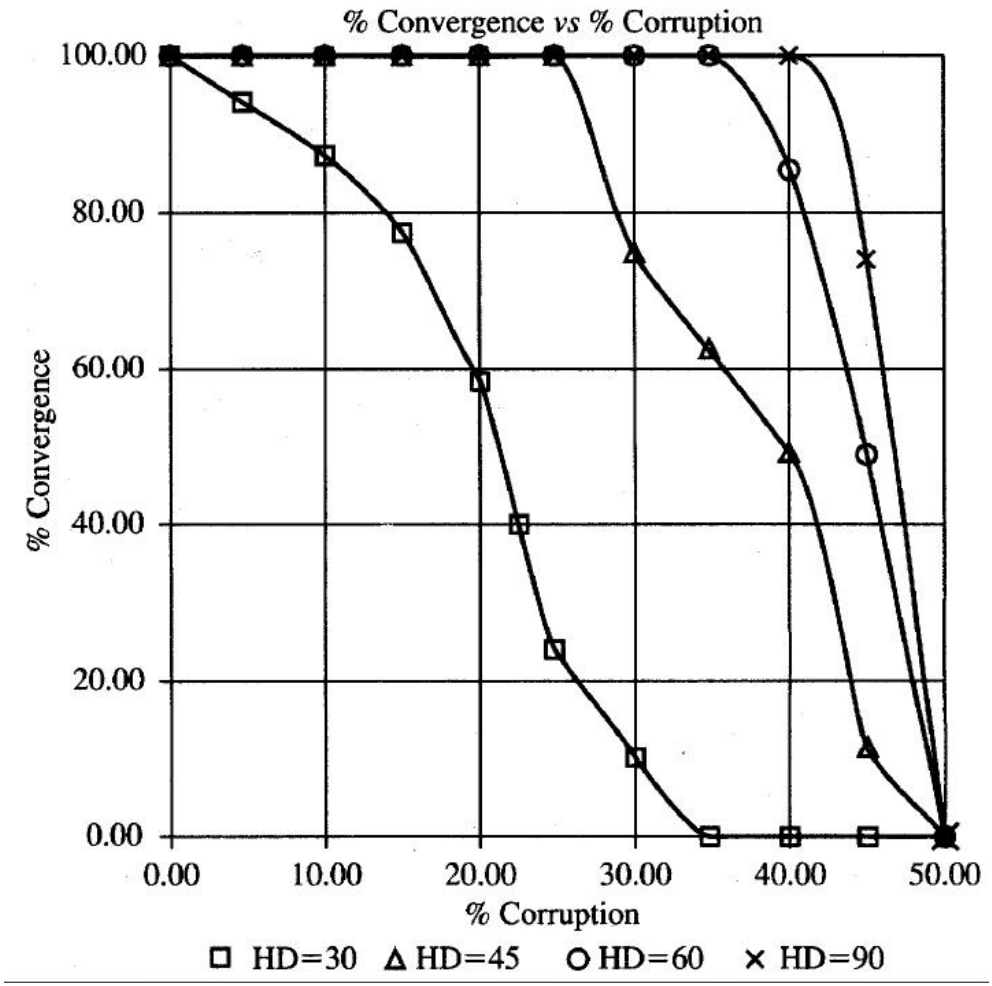
2. If the stored patterns are not orthogonal(Statistically they are not fully independent):
   - $\eta \neq 0$
   - η may affect the sign of *net*
   - The noise term
     increases for an increased number of stored patterns
     becomes relatively significant when the factor $(n - p)$ decreases

Memory Convergence versus Corruption

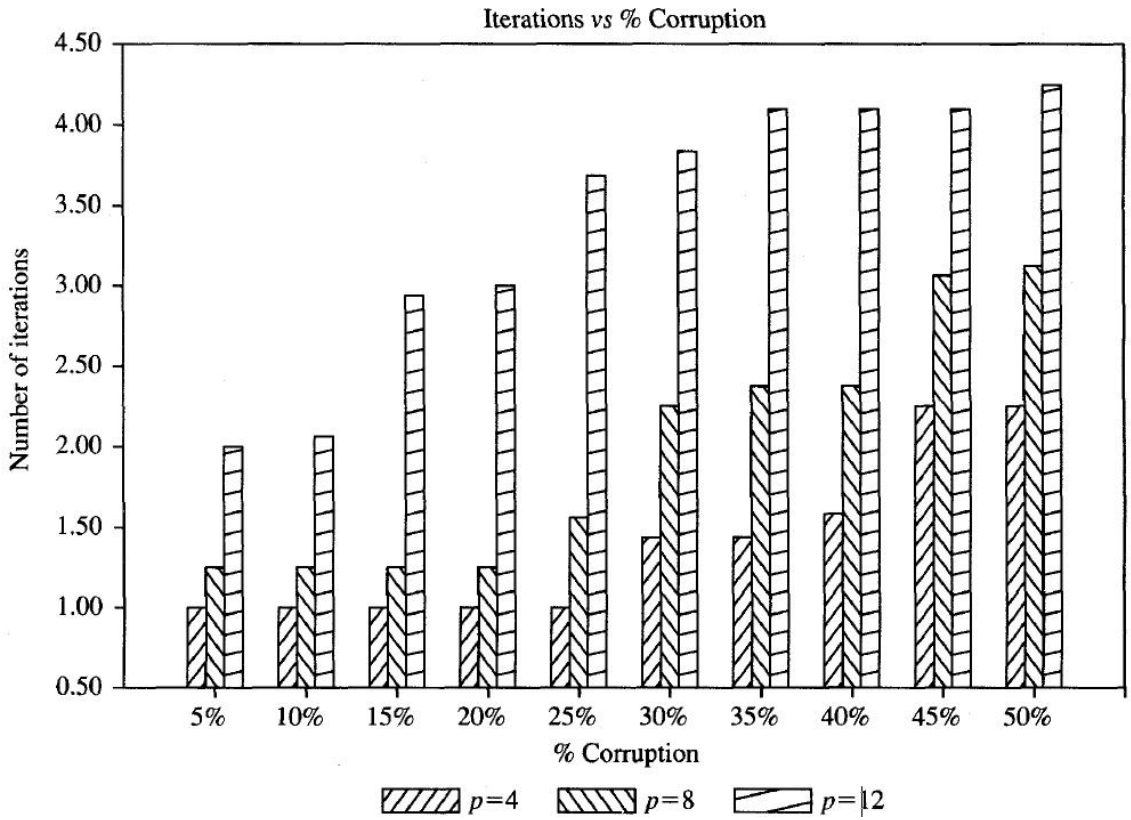
- Assume $n = 120$, HD of stored vectors is fixed and equal to 45
- The correct convergence rate drops about linearly with the amount of corruption of the key vector.
- The correct convergence rate also reduces as the number of stored patterns increases for a fixed distortion value of input key vectors.
- The network performs very well at $p = 2$ patterns stored
- It recovers rather poorly distorted vectors at $p = 16$ patterns stored.



- Assume $n = 120$, P is fixed and equal to 4
- The network exhibits high noise immunity for large and very large Hamming distances between the stored vectors.
- For stored vectors that have 75% of the bits in common, the recovery of correct memories is shown to be rather inefficient.



- The average number of measured update cycles has been between 1 and 4.
- This number increases almost linearly with the number of patterns stored and with the percent corruption of the key input vector.

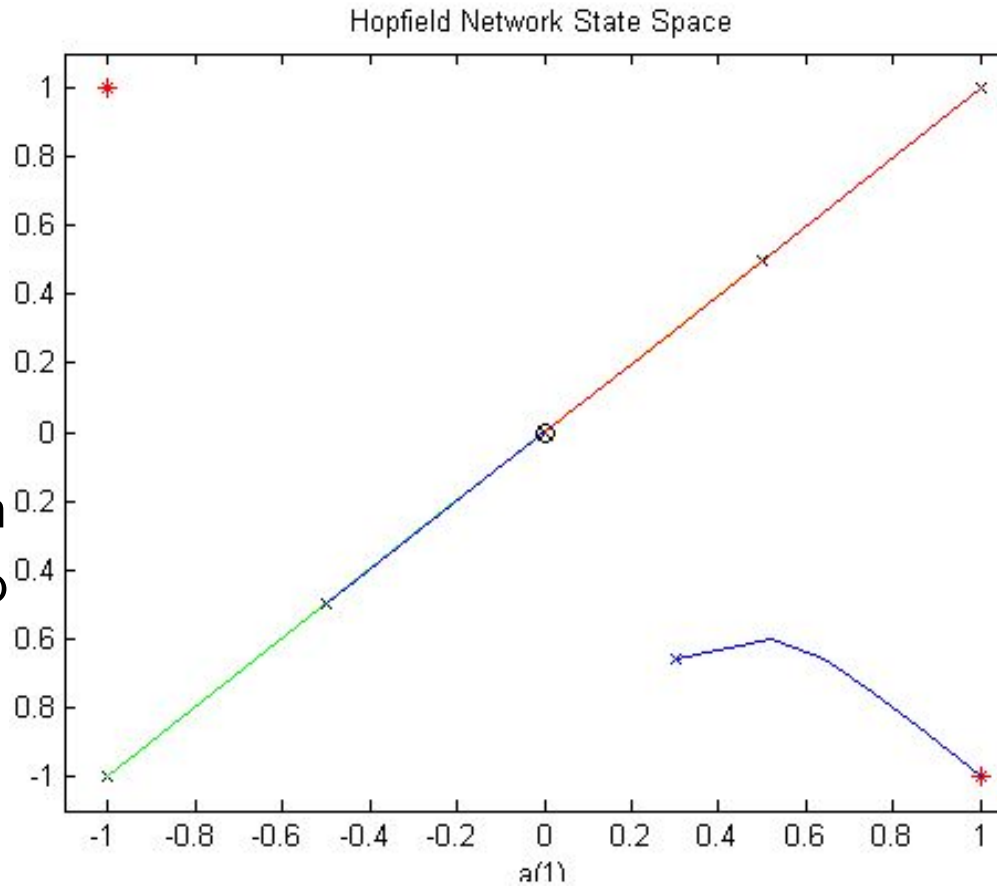


Problems

- Uncertain Recovery
  - Heavily overloaded memory ($p/n > 50\%$) may not be able to provide error-free or efficient recovery of stored pattern.
  - Theres are some examples of convergence that are not toward the closest memory as measured with the HD value
- Undesired Equilibria
  - Spurious points are stable equilibria with minimum energy that are additional to the patterns we already stored.
  - The undesired equilibria may be unstable. Therefore, any noise in the system will move the network out of them.

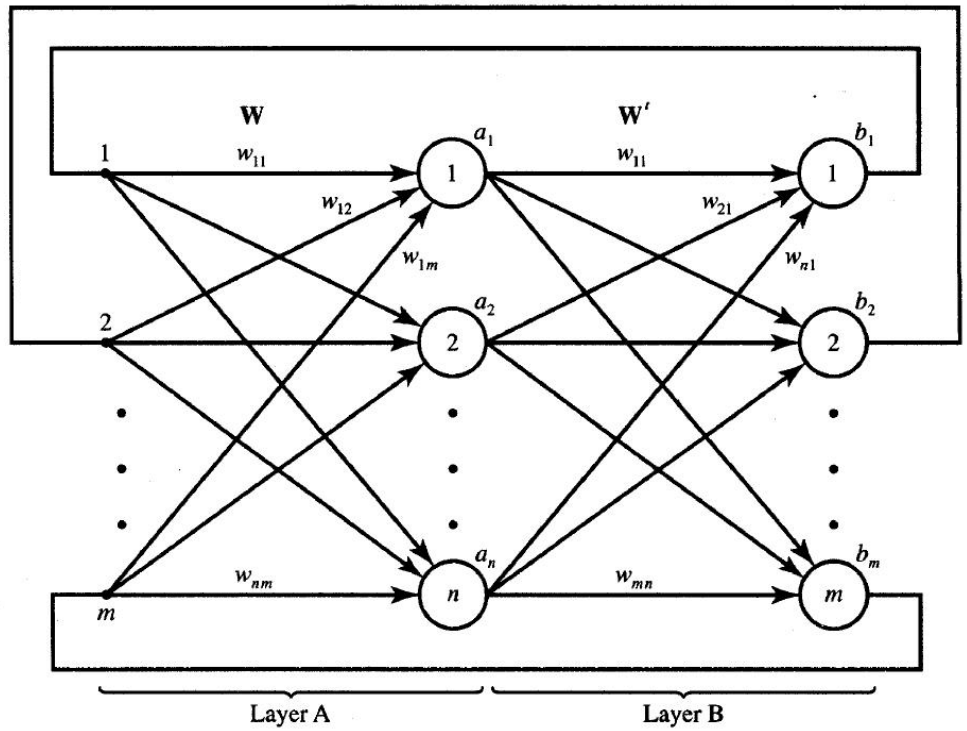
Example

- Assume two patterns are stored in the memory as shown in Fig.
- The input vector converges to the closer pattern
- BUT if the input is exactly between the two stable points, it moves into the center of the state space!



Bidirectional Associative Memory (Kosko 1987, 1988)

- It is a heteroassociative, content-addressable memory consisting of two layers.
- It uses the forward and backward information flow to produce an associative search for stored stimulus-response association
- The stability corresponds to a local energy minimum.
- The network's dynamics involves two layers of interaction.
- The patterns are $\{(a^{(1)}, b^{(1)}), (a^{(2)}, b^{(2)}), \ldots, (a^{(p)}, b^{(p)})\}$



Memory Architecture

- Assume an initializing vector b is applied at the input (layer A)
- The neurons are assumed to be bipolar binary.

$$\begin{aligned} a' &= \Gamma[Wb] \\ a'_i &= sgn(\sum_{j=1}^{m} w_{ij} b_j), \ i = 1, ..., n \end{aligned}$$

- $\Gamma[.]$ is a nonlinear activation function. For discrete-time networks, it is usually a TLU
- Now vector a' is applied to Layer B

$$\begin{aligned} b' &= \Gamma[W' a'] \\ b'_j &= sgn(\sum_{i=1}^{n} w_{ij} a_i), \ j = 1, ..., m \end{aligned}$$

- So the procedure can be shown as

$$\begin{array}{cc} \text{First Forward Pass} & a^1 = \Gamma[Wb^0] \\ \text{First backward Pass} & b^2 = \Gamma[W^T a^1] \\ \text{Second Forward Pass} & a^3 = \Gamma[Wb^2] \\ \vdots & \vdots \\ \text{k/2 th backward Pass} & b^k = \Gamma[W^T a^{(k-1)}] \end{array} \quad (1)$$

- If the network is stable the procedure is stope at an equilibrium pair like $(a^{(i)}, b^{(i)})$
- Both asynchronously and synchronously updating yields stability.
- To achieve faster result it is preferred to update synchronously
- If the weight matrix is square and symmetric, $W = W^T$, then both memories become identical and autoassociative.

- **Storage Algorithm** :
  - It is based on Hebbian rule:

$$W = \sum_{i=1}^{p} a^{(i)} b^{(i)T}$$

$$w_{ij} = \sum_{m=1}^{P} a_i^{(m)} b_j^{(m)}$$

  - $a^{(i)}$ and $b^{(i)}$ are bipolar binary
- **Stability Analysis**
  - bidirectionally stable means that the updates continue and the memory comes to its equilibrium at the k^{th} step, we have $a^k \longrightarrow b^k \longrightarrow a^{k+2}$, and $a^{k+2} = a^k$
  - Consider the energy function $E(a, b) = -\frac{1}{2}(a^T W b) - \frac{1}{2}(b^T W^T a)$
  - Since a and b are vectors $E(a, b) = -(a^T W b)$

- The gradients of energy with respect to a and b:

$$\begin{aligned} \nabla_a E(a,b) &= -Wb \\ \nabla_b E(a,b) &= -W^T a \end{aligned}$$

- Energy changes due to the single bit increments $\triangle a_i$ and $\triangle b_j$:

$$\begin{aligned} \triangle E_{a_i} &= -(\sum_{j=1}^{m} w_{ij} b_j) \triangle a_i \;\; i = 1, ..., n \\ \triangle E_{b_i} &= -(\sum_{i=1}^{n} w_{ij} a_i) \triangle b_j \;\; j = 1, ..., m \end{aligned}$$

- Considering the output update law (1):

$$\triangle a_i = \begin{cases} -2 & \sum_{j=1}^{m} w_{ij} b_j < 0 \\ 0 & \sum_{j=1}^{m} w_{ij} b_j = 0 \\ 2 & \sum_{j=1}^{m} w_{ij} b_j > 0 \end{cases}$$

$$\triangle b_j = \begin{cases} -2 & \sum_{i=1}^{n} w_{ij} a_i < 0 \\ 0 & \sum_{i=1}^{n} w_{ij} a_i = 0 \\ 2 & \sum_{i=1}^{n} w_{ij} a_i > 0 \end{cases}$$

- $\therefore \triangle E \leq 0$
- and E is bounded : $E \geq -\sum_{i=1}^{N} \sum_{j=1}^{M} |w_{ij}| \rightsquigarrow$ the memory converges to a stable point

Example

- Consider four 16-pixel bit maps of letter characters should be associated to 7-bit binary vectors as below:

$$
\begin{aligned}
a^{(1)} &= [1\ 1\ 1\ -1\ 1\ -1\ -1\ -1\ 1\ -1\ -1\ -1\ 1\ 1\ 1\ -1]^T, \\
b^{(1)} &= [1\ -1\ -1\ -1\ -1\ 1\ 1]^T, \\
a^{(2)} &= [1\ -1\ -1\ 1\ 1\ 1\ -1\ 1\ 1\ -1\ 1\ 1\ 1\ -1\ -1\ 1]^T, \\
b^{(2)} &= [1\ -1\ -1\ 1\ 1\ 1\ -1]^T, \\
a^{(3)} &= [1\ 1\ 1\ 1\ -1\ -1\ 1\ -1\ -1\ 1\ -1\ -1\ 1\ 1\ 1\ 1]^T, \\
b^{(3)} &= [1\ -1\ 1\ 1\ -1\ 1\ -1]^T, \\
a^{(4)} &= [-1\ -1\ -1\ -1\ -1\ -1\ -1\ 1\ 1\ 1\ 1\ 1\ -1\ -1\ -1\ 1]^T, \\
b^{(4)} &= [-1\ 1\ 1\ -1\ 1\ -1\ 1]^T,
\end{aligned}
$$

- $W = \sum_{i=1}^{p} a^{(i)} b^{(i)T}$

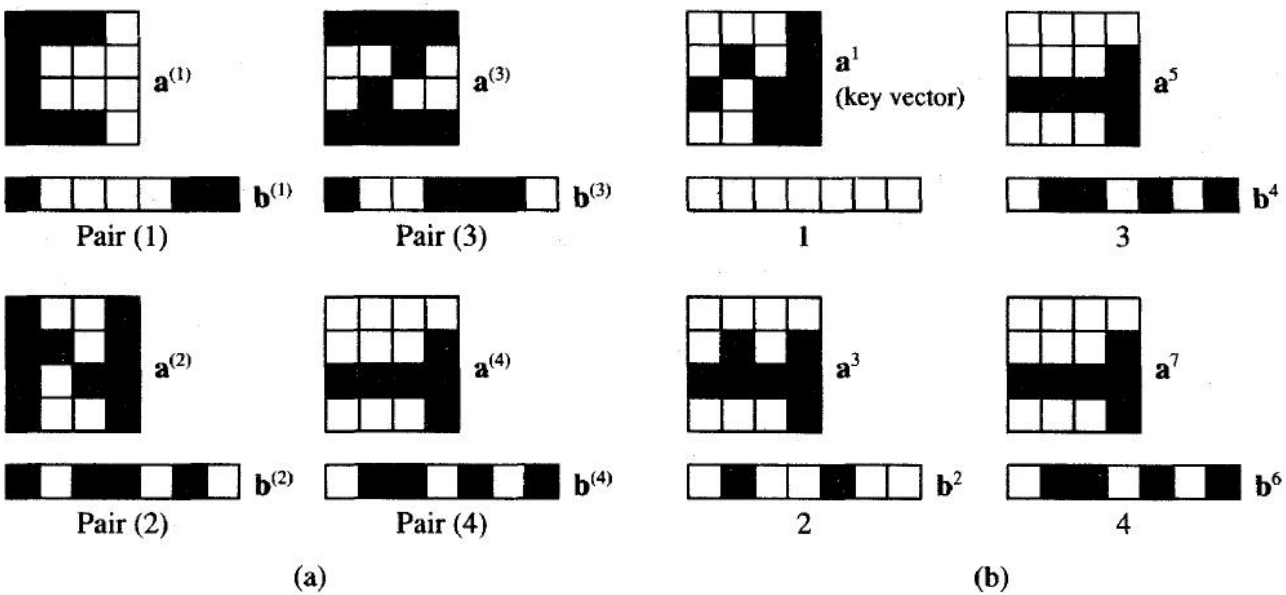
- Assume a key vector a^1 at the memory input is a distorted prototype of $a^{(2)}$, $HD(a^{(2)}, a^1) = 4$:

$$a^1 = [-1\ -1\ -1\ 1\ -1\ 1\ -1\ 1\ 1\ -1\ 1\ 1\ -1\ -1\ 1\ 1]^T,$$

- Therefore:

$$\begin{aligned}
b^2 &= \Gamma[-16\ 16\ 0\ 0\ 32\ -16\ 0]^T = [-1\ 1\ 0\ 0\ 1\ -1\ 0]^T,\\
a^3 &= [-1\ -1\ -1\ -1\ -1\ 1\ -1\ 1\ 1\ 1\ 1\ 1\ 1\ -1\ -1\ 1]^T,\\
b^4 &= [-1\ 1\ 1\ -1\ 1\ -1\ 1]^T,\\
a^5 &= [-1\ -1\ -1\ -1\ -1\ -1\ -1\ 1\ 1\ 1\ 1\ 1\ 1\ -1\ -1\ -1\ 1]^T,\\
b^6 &= b^4,\ a^7 = a^5
\end{aligned}$$

- $HD(a^{(1)}, a^1) = 12, HD(a^{(2)}, a^1) = 4,\ HD(a^{(3)}, a^1) = 10$,$HD(a^{(4)}, a^1) = 4$



Example of bidirectional associative memory: (a) associations stored and (b) four steps of retrieval of pair 4.

Multidimensional Associative Memory (MAM)

- Bidirectional associative memory is a two-layer nonlinear recurrent network for stored stimulus-response associations $(a^{(i)}, b^{(i)}), i = 1, 2, ..., p$.
- The bidirectional model can be generalized to multiple associations $(a^{(i)}, b^{(i)}, c^{(i)}, ...),\ i = 1, 2, ..., p$. (Hagiwara 1990)
- For n tuple association memory, n layer network should be considered.
- Layers are interconnected with each other by weights that pass information between them.
- For a triple association memory:

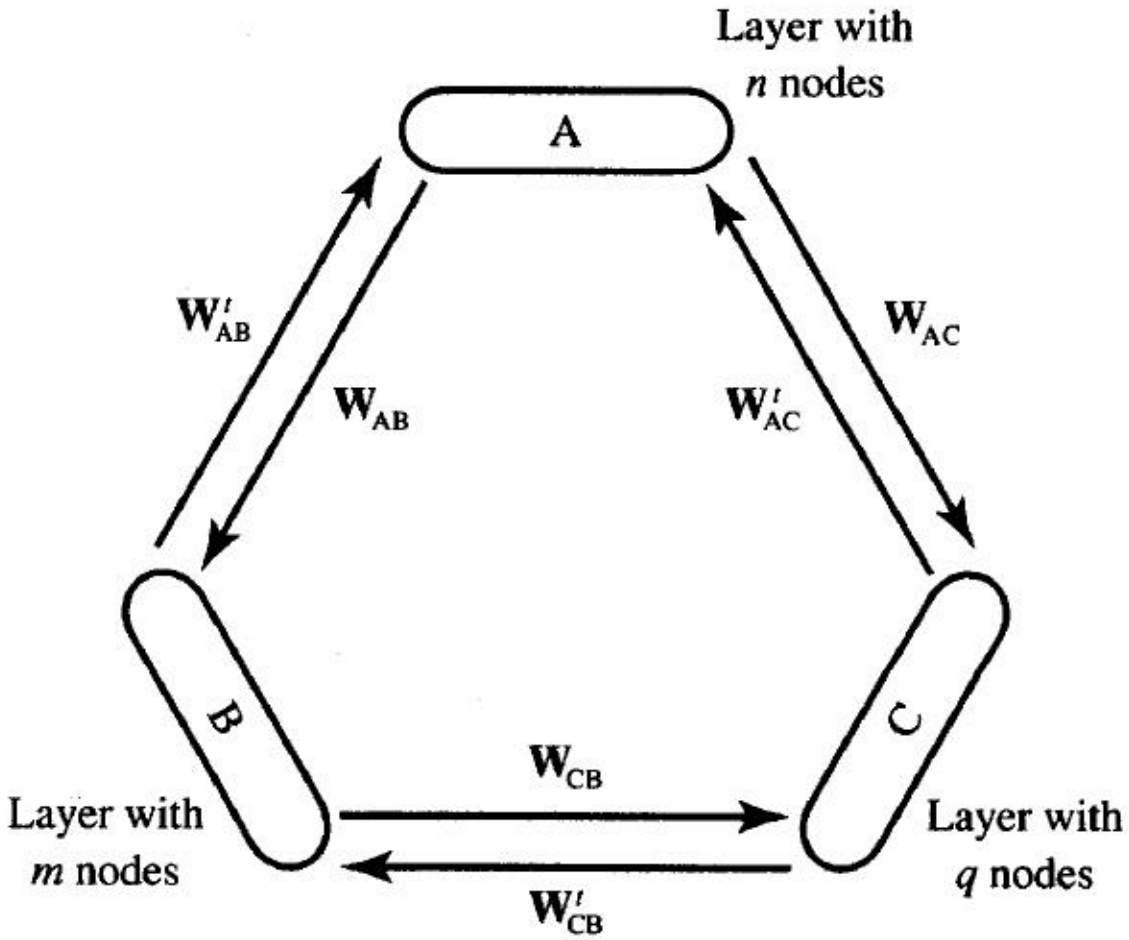
$$W_{AB} = \sum_{i=1}^{p} a^{(i)} b^{(i)T}, W_{CB} = \sum_{i=1}^{p} c^{(i)} b^{(i)T}$$

$$W_{AC} = \sum_{i=1}^{p} a^{(i)} c^{(i)T}$$

- Each neuron independently and synchronously updates its output based on its total input sum from all other layers:

$$\begin{aligned} a' &= \Gamma[W_{AB}b + W_{AC}c] \\ b' &= \Gamma[W_{CB}^T c + W_{AB}^T a] \\ c' &= \Gamma[W_{AC}^T a + W_{CB}b] \end{aligned}$$

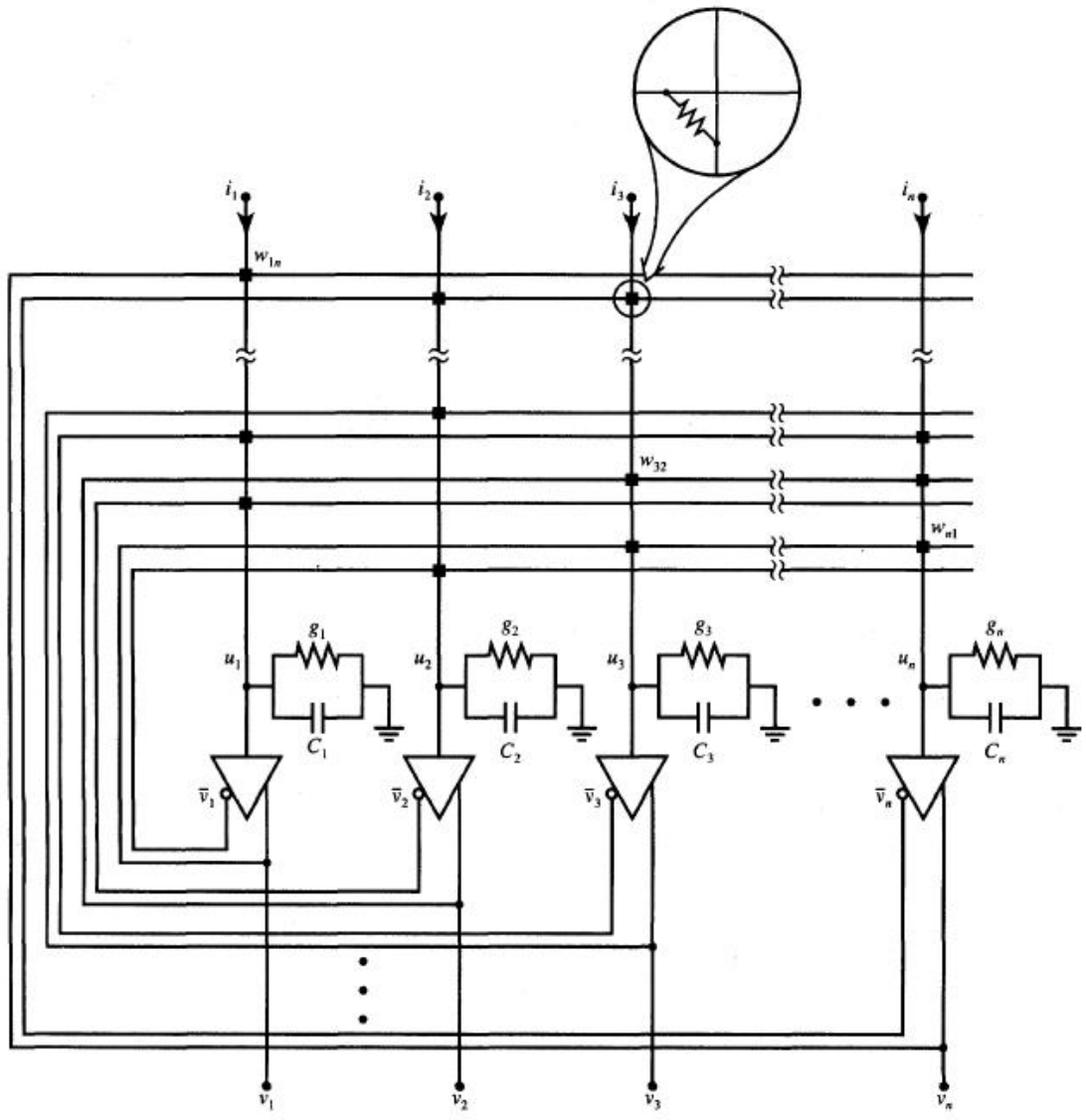
- The neurons' states change synchronously until a multidirectionally stable state is reached



Gradient-Type Hopfield Network

- Gradient-type neural networks are generalized Hopfield networks in which the computational energy decreases continuously in time.
- Gradient-type networks converge to one of the stable minima in the state space.
- The evolution of the system is in the general direction of the negative gradient of an energy function.
- Typically, the network energy function is equivalent to a certain objective (penalty) function to be minimized.
- These networks are examples of nonlinear, dynamical, and asymptotically stable systems.
- They can be considered as a solution of an optimization problem.

- The model of a gradient-type neural system using electrical components is shown in Fig.
- It has n neurons,
- Each neuron mapping its input voltage u_i into the output voltage v_i through the activation function $f(u_i)$,
- $f(u_i)$ is the common static voltage transfer characteristic (VTC) of the neuron.



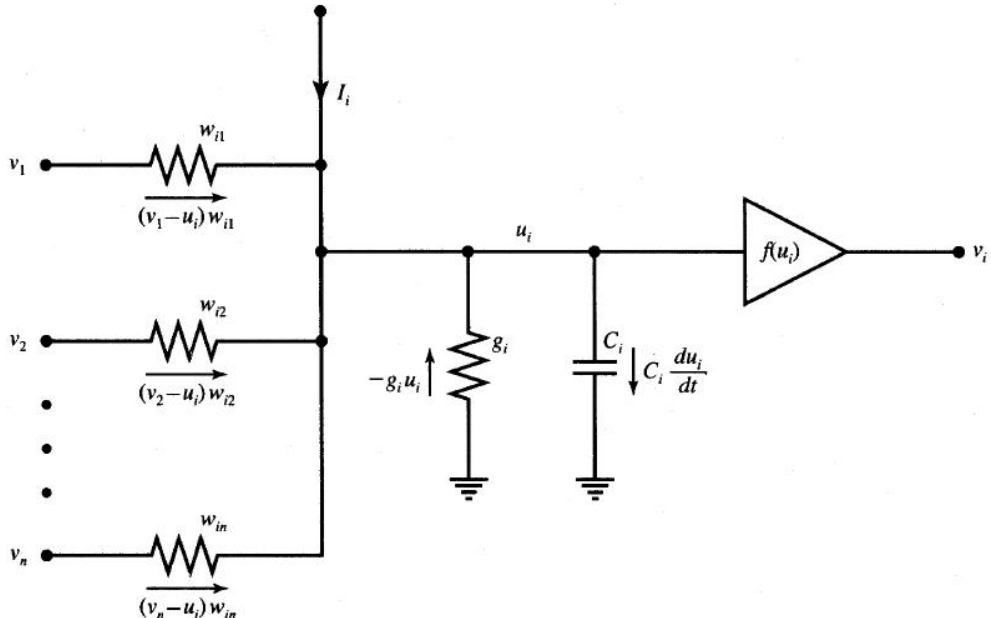
- Conductance w_{ij} connects the output of the jth neuron to the input of the ith neuron.
- The inverted neuron outputs $\bar{v}_i$ representing inverting output is applied to avoid negative conductance values w_{ij}
- Note that in Hopefield networks:
  - $w_{ij} = w_{ji}$
  - $w_{ii} = 0 \rightsquigarrow$, the outputs of neurons are not connected back to their own inputs
- Capacitances C_i, for $i = 1, 2, ..., n$, are responsible for the dynamics of the transients in this model.

- KCL equ. for each node is

$$i_i + \sum_{j \neq i}^{n} W_{ij} v_j - u_i \left(\sum_{j \neq i}^{n} w_{ij} + g_i \right) = C_i \frac{du}{dt} \tag{2}$$

- Considering $G_i = \sum_{j=1}^{n} w_{ij} + g_i$, $C = diag[C_1, C_2, ..., C_n]$, $G = [G_1, ..., G_n]$, the output equ. for whole system is

$$\begin{aligned} C\frac{du}{dt} &= Wv(t) - Gu(t) + I \\ v(t) &= f[u(t)] \end{aligned} \tag{3}$$

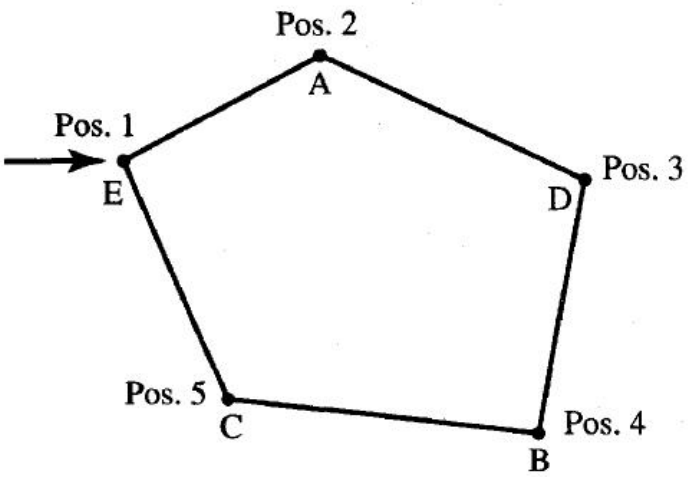


- The energy to be minimized is $E(v) = -\frac{1}{2} v^t W v - iv + \frac{1}{\lambda} \sum_{i=1}^{n} G_i \int_0^{v_i} f_i^{-1}(z) dz$
- To study the stability one should check the sign of $\frac{dE[v(t)]}{dt}$
- $\frac{dE[v(t)]}{dt} = \nabla E^t(v) \dot{v}; \nabla E^t(v) = [\frac{\partial E(v)}{\partial v_1} \quad \frac{\partial E(v)}{\partial v_2} \quad \cdots \frac{\partial E(v)}{\partial v_n}]$
- $\frac{d}{d_{v_i}}(G_i \int_0^{v_i} f_i^{-1}(z) dz) = G_i u_i \rightsquigarrow \frac{dE[v(t)]}{dt} = (-Wv - i + Gu)^t \dot{v}$
- Now considering (3) yields: $\frac{dE[v(t)]}{dt} = -(C \frac{du}{dt})^t \frac{dv}{dt}$
- Using the inverse activation function $f^{-1}(v_i)$ and chain rule leads to $C_i \frac{du_i}{dt} = C_i f^{-1'}(v_i) \frac{dv_i}{dt}$
- Since $f^{-1'}(v_i) > 0$, $\frac{dv_i}{dt}$ and $\frac{du_i}{dt}$ have the same sign
- $\therefore$ The change of E in time, $(\frac{dE[v(t)]}{dt})$, is toward lower values of energy.

- When $\frac{dE[v(t)]}{dt} = 0 \rightsquigarrow \frac{du_i}{dt} = 0$ for $i = 1, ..., n$
- Therefore, there is no oscillation and cycle, and the updates stop at minimum energy.
- The Hopfield networks can be applied for optimization problems.
- The challenge will be defining W and I s.t. fit the dynamics and objective of the problem to (3) and above equation.

Example: Traveling Salesman Tour Length [?]

- The problem is min tour length through a number of cities with only one visit of each city
- If n is number of cities $(n-1)!$ distinct path exists
- Let us use Hopefiled network to find the optimum solution
- We are looking to find a matrix shown in the fig.
  - n rows are the cities
  - n columns are the position of the salesman
  - each city/position can take 1 or 0
  - $v_{ij} = 1$ means salesman in its jth position is in ith city
- The network consists n^2 unipolar neurons
- Each city should be visited once $\rightsquigarrow$ only one single 1 at each row and column



City, y						
A	0	1	0	0	0	5
B	0	0	0	1	0	4
C	0	0	0	0	1	3
D	0	0	1	0	0	2
E	1	0	0	0	0	1
	1	2	3	4	5	

Position, x

- We should define w and i such that the energy of the Hopfield network represent the objective we are looking for
- Recall the energy of Hoefiled network:
  $E(v) = -\frac{1}{2}\sum_{Xi}\sum_{Yj} w_{Xi,Yj} v_{Xi} v_{Yj} - \sum_{Xi} i_{Xi} v_{Xi}$
  - The last term is omitted for simplicity
- Let us express our objective in math:
  $E_1 = A\sum_X \sum_i \sum_j v_{Xi} v_{Xj}$ for $i \neq j$
  $E_2 = B\sum_i \sum_X \sum_Y v_{Xi} v_{Yi}$ for $X \neq Y$
- E_1 be zero $\rightsquigarrow$ each row has at most one 1
- E_2 be zero $\rightsquigarrow$ each column has at most one 1
- $E_3 = C(\sum_X \sum_i v_{Xi} - n)^2$
- E_3 guarantees that there is at least one 1 at each column and row.
- $E_4 = D\sum_X \sum_Y \sum_i d_{XY} v_{Xi}(v_{Y,i+1} + v_{Y,i-1}), \quad X \neq Y$
- E_4 represents minimizing the distances
- d_{XY} is distance between city X and Y

- Recall the energy of Hoefiled network:
  $E(v) = -\frac{1}{2}\sum_{Xi}\sum_{Yj} w_{Xi,Yj} v_{Xi} v_{Yj} - \sum_{Xi} i_{Xi} v_{Xi}$
- The weights can be defined as follows
- $\therefore$
  $W_{Xi,Yj} = -2A\delta_{XY}(1-\delta_{ij}) - 2B\delta_{ij}(1-\delta_{XY}) - 2C - 2Dd_{xy}(\delta_{j,i+1}+\delta_{j,j-1})$
- $i_{Xi} = 2Cn$
- Positive consts A, B, C, and D are selected heuristically