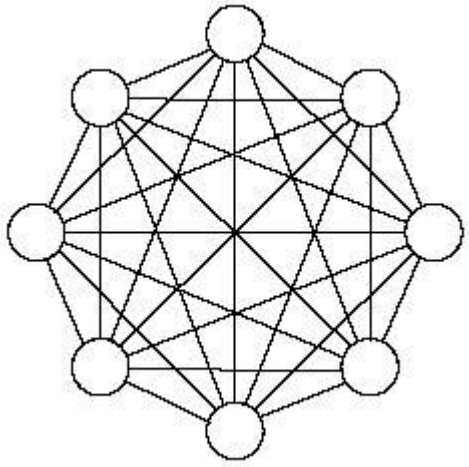
Hopfield network

Presenter: Seok Ho-Sik

Reference: http://www.comp.leeds.ac.uk/ai23/reading/Hopfield.pdf

Introduction (1/2)

- A Hopfield network is a form of recurrent artificial neural network invented by John Hopfield.
- Hopfield networks is regarded as a helpful tool for understanding human memory.
- Hopfield networks serve as content addressable memory systems with binary threshold units.



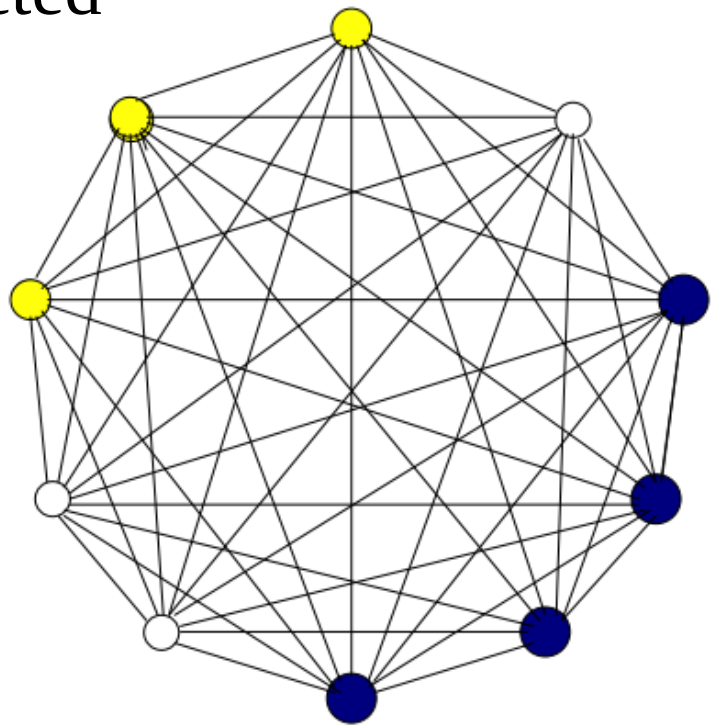
Introduction (2/2)

- It can be used as associative memory.
  - Associative memory: a memory that is addressable through its contents.
  - If a pattern is presented to an associative memory, it returns whether this pattern coincides with a stored pattern.
  - This memory can return a stored pattern that is similar to the presented one.

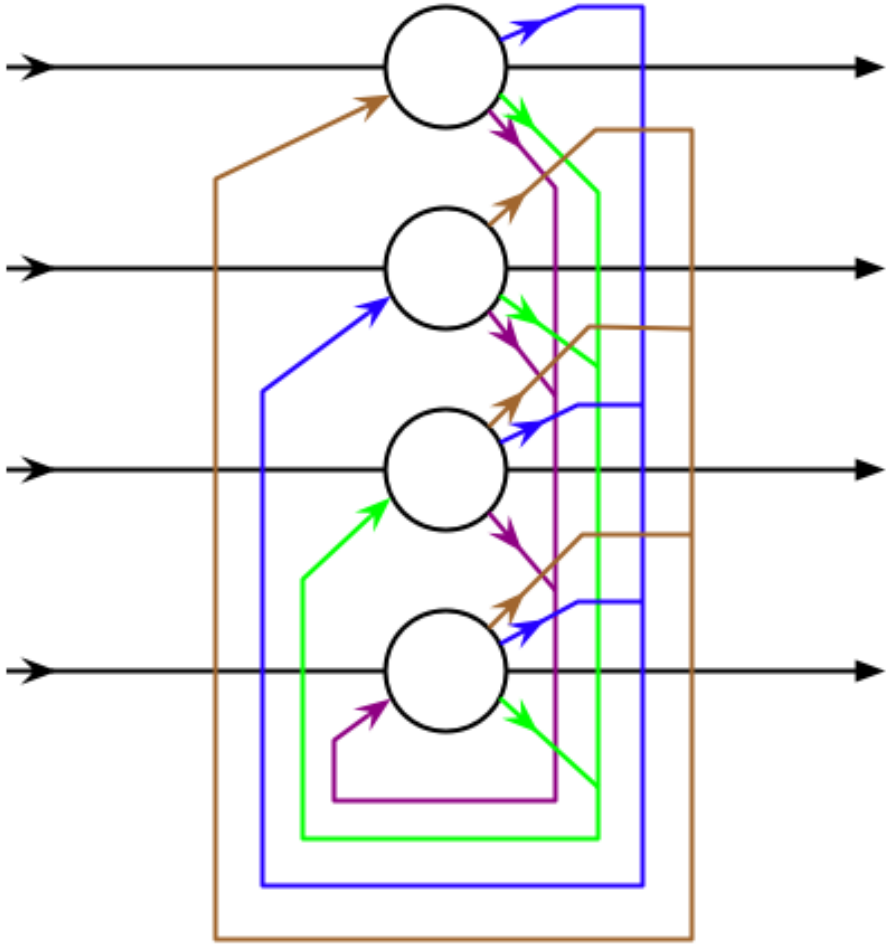
Structure (1/2)

- Hopfield networks are constructed from artificial neurons
  - A network of fully connected N artificial neurons.
  - N inputs, weight w_i, an output.

$$o = \begin{cases} 1 & : \quad \sum_i w_i x_i \geq 0 \\ -1 & : \quad \sum_i w_i x_i < 0 \end{cases}$$



- Updating rule
  - To determine the value of each input.
  - To calculate the weighted sum of all inputs.
  - To determine the output state.



Structure (2/2)

- Updating rule
  - Asynchronous: one picks one neuron, calculates the weighted input sum and updates immediately.
  - Synchronous: the weighted input sums of all neurons are calculated without updating the neurons. Then all neurons are set to their new value.

Network training – example

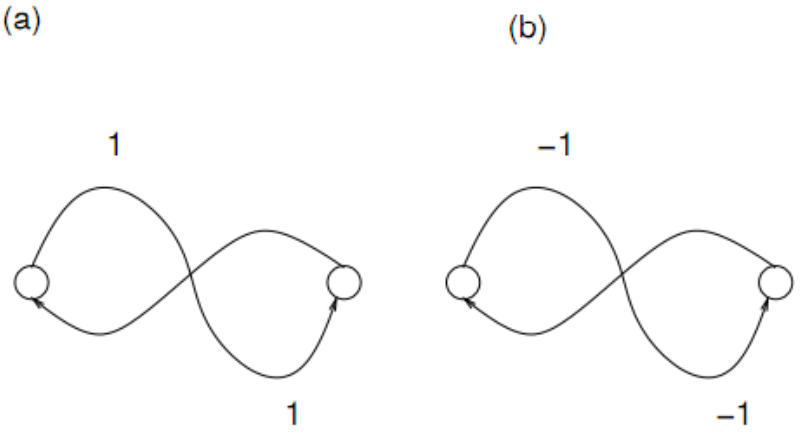


Figure 3: A two-neuron network. There are just two options for the weight matrix: $w_{ij} = 1$ (A) or $w_{ij} = -1$. If the weight is 1, there are two stable states under synchronous updating $\{+1, +1\}$, or $\{-1, -1\}$. For a weight of -1, the stable states will be $\{-1, +1, \}$ or $(+1, -1\}$ depending on initial conditions. Under synchrononous updating the states are oscillatory.

- The steady final state is determined by the value of the weight.
- Under synchronous updating, there are no stable states in the network → stable states are achieved under asynchronous updating.

Usage

- A pattern is entered in the network by setting all nodes to a specific value, or by setting only part of the nodes → the network is subject to a number of iterations using asynchronous or synchronous updating.
- Content Addressable Memory
  - Patterns are stored in the weight matrix → the dynamics of the network then retrieve the patterns.
- Cue and association
  - By entering the cue, the entire pattern is retrieved.
  - In this way, the network restores the association that belongs to a given cue.

Weight matrix and energy

- Constraints
  - Symmetry: $w_{ij} = w_{ji}$
  - No self connections: $w_{ii} = 0$
- Given a network of N nodes and faced with a pattern $\vec{x} = \{x_1,,,x_N\}$ that we want to store in the network, then

$$w_{ij} = x_i x_j$$

- Energy
  - $h_i \equiv \sum_j w_{ij} x_j$: the weighted input sum of a node (local field)
  - $E_i = -\frac{1}{2} h_i x$
  - For the entire network

$$E(\vec{x}) = \sum_i E_i = -\sum_i \frac{1}{2} h_i x_i = -\frac{1}{2} \sum_{ij} w_{ij} x_i x_j$$

Stability of a single pattern

- Situation: a pattern in the network ($\vec{y}$) is different from the input pattern ($\vec{x}$).
- For a network with N nodes, updating a node i of the network. $h_i = \sum_j w_{ij} y_j = \sum_j x_i x_j y_j$
- If $\vec{x}$ and $\vec{y}$ are different in k points (having opposite sign) then $h_i = \sum_{j=1}^{N-k} x_i x_j x_j + \sum_{j=1}^{k} x_i x_j (-x_j) = (N - 2K) x_i$

  therefore, updating will result in no change when $N>2k$.

The Hebb rule and the generalized Hebb rule

- How to store more than one single pattern in the network?
  - Usual notation of weight $w_{ij} = \frac{1}{N} x_i x_j$
  - If we want to store two patterns $\vec{x}^{(1)}$ & $\vec{x}^{(2)}$
    - Calculate weight for each pattern separately

      $w_{ij}^{(1)} = \frac{1}{N} x_i^{(1)} x_j^{(1)}$ and $w_{ij}^{(2)} = \frac{1}{N} x_i^{(2)} x_j^{(2)}$

      then $w_{ij} = w_{ij}^{(1)} + w_{ij}^{(2)}$
    - For p patterns $w_{ij} = \frac{1}{N} \sum_{k=1}^{p} x_i^{(k)} x_j^{(k)}$
- If the number of node N is greater than the number of patterns p ($p << N$), it is highly likely that such a energy minimum will correspond to one of the training patterns.

Exercise

- http://www.cbu.edu/~pong/ai/hopfield/hopfieldapplet.html
- Observe the behavior of the Hopfield network in the various settings.
  - Change the "Grid size"
  - Change the "Corruption %"
  - Etc.

Homework

- Due: Sep. 27th
- Hardcopy submission to TA.
- Problem 1
  - http://www.cbu.edu/~pong/ai/hopfield/hopfieldapplet.html
  - Experiment with various “corruption %” and “Stored patterns”.
  - Discuss how much noise the Hopfield network can tolerate.
- Problem 2
  - For the Hopfield network with 4 neurons (each neuron can take the values -1 or +1)

  a. Calculate the weights needed to store the pattern (-1,1,-1,1)

  b. Using the weights you calculated, determine if the pattern (-1, 1, -1, 1) is stable.