

Computational Intelligence

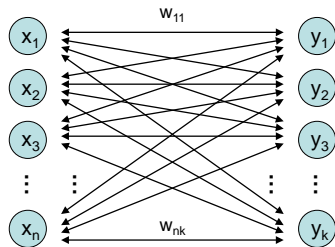
Winter Term 2010/11

Prof. Dr. Günter Rudolph
 Lehrstuhl für Algorithm Engineering (LS 11)
 Fakultät für Informatik
 TU Dortmund

- Bidirectional Associative Memory (BAM)
 - Fixed Points
 - Concept of Energy Function
 - Stable States = Minimizers of Energy Function

- Hopfield Network
 - Convergence
 - Application to Combinatorial Optimization

Network Model



- fully connected
- bidirectional edges
- synchronized:
 - step t : data flow from x to y
 - step t + 1 : data flow from y to x

start: $y^{(0)} = \text{sgn}(x^{(0)} W)$
 $x^{(1)} = \text{sgn}(y^{(0)} W')$
 $y^{(1)} = \text{sgn}(x^{(1)} W)$
 $x^{(2)} = \text{sgn}(y^{(1)} W')$
 ...

x, y : row vectors
 W : weight matrix
 W' : transpose of W
 bipolar inputs $\in \{-1, +1\}$

Fixed Points

Definition

(x, y) is **fixed point** of BAM iff $y = \text{sgn}(x W)$ and $x' = \text{sgn}(W y')$. □

Set $W = x' y$. (note: x is row vector)

$$y = \text{sgn}(x W) = \text{sgn}(x (x' y)) = \text{sgn}(\underbrace{(x x')}_{> 0} y) = \text{sgn}(\|x\|^2 y) = y$$

> 0 (does not alter sign)

$$x' = \text{sgn}(W y) = \text{sgn}((x' y) y') = \text{sgn}(x' \underbrace{(y y')}_{> 0}) = \text{sgn}(x' \|y\|^2) = x'$$

> 0 (does not alter sign)

Theorem: If $W = x' y$ then (x, y) is fixed point of BAM. □

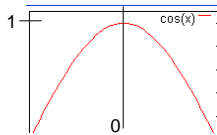
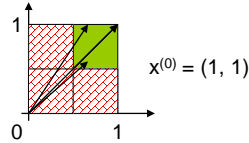
Concept of Energy Function

given: BAM with $W = x'y$ $\Rightarrow (x,y)$ is stable state of BAM

starting point $x^{(0)}$ $\Rightarrow y^{(0)} = \text{sgn}(x^{(0)} W)$
 \Rightarrow excitation $e' = W(y^{(0)})'$
 \Rightarrow if $\text{sign}(e') = x^{(0)}$ then $(x^{(0)}, y^{(0)})$ stable state

small angle between e' and $x^{(0)}$

\Leftarrow true if e' close to $x^{(0)}$



small angle $\alpha \Rightarrow$ large $\cos(\alpha)$

recall: $\frac{ab'}{\|a\| \cdot \|b\|} = \cos \angle(a, b)$

Concept of Energy Function

required:

small angle between $e' = W y^{(0) '}$ and $x^{(0)}$

\Rightarrow larger cosine of angle indicates greater similarity of vectors

$\Rightarrow \forall e'$ of equal size: try to maximize $x^{(0)} e' = \underbrace{\|x^{(0)}\|}_{\text{fixed}} \cdot \underbrace{\|e'\|}_{\text{fixed}} \cdot \underbrace{\cos \angle(x^{(0)}, e')}_{\rightarrow \text{max!}}$

\Rightarrow maximize $x^{(0)} e' = x^{(0)} W y^{(0) '}$

\Rightarrow identical to minimize $-x^{(0)} W y^{(0) '}$

Definition

Energy function of BAM at iteration t is $E(x^{(t)}, y^{(t)}) = -\frac{1}{2} x^{(t)} W y^{(t) '}$ \square

Stable States

Theorem

An asynchronous BAM with arbitrary weight matrix W reaches steady state in a finite number of updates.

Proof:

$$E(x, y) = -\frac{1}{2} x W y' = \begin{cases} -\frac{1}{2} x (W y') = -\frac{1}{2} x b' = -\frac{1}{2} \sum_{i=1}^n b_i x_i \\ -\frac{1}{2} (x W) y' = -\frac{1}{2} a y' = -\frac{1}{2} \sum_{i=1}^k a_i y_i \end{cases}$$

$\swarrow \searrow$ excitations

BAM asynchronous \Rightarrow select neuron at random from left or right layer, compute its excitation and change state if necessary (states of other neurons not affected)

neuron i of left layer has changed $\Rightarrow \text{sgn}(x_i) \neq \text{sgn}(b_i)$

$\Rightarrow x_i$ was updated to $\tilde{x}_i = -x_i$

$E(x, y) - E(\tilde{x}, y) = -\frac{1}{2} \underbrace{b_i (x_i - \tilde{x}_i)}_{< 0} > 0$

x_i	b_i	$x_i - \tilde{x}_i$
-1	> 0	< 0
+1	< 0	> 0

use analogous argumentation if neuron of right layer has changed

\Rightarrow every update (change of state) decreases energy function

\Rightarrow since number of different bipolar vectors is finite update stops after finite #updates

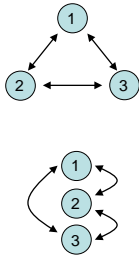
remark: dynamics of BAM get stable in local minimum of energy function!

q.e.d.

special case of BAM but proposed earlier (1982)

characterization:

- neurons preserve state until selected at random for update
- n neurons fully connected
- symmetric weight matrix
- no self-loops (\rightarrow zero main diagonal entries)
- thresholds θ , neuron i fires if excitations larger than θ_i



transition: select index k at random, new state is $\tilde{x} = \text{sgn}(xW - \theta)$

where $\tilde{x} = (x_1, \dots, x_{k-1}, \tilde{x}_k, x_{k+1}, \dots, x_n)$

energy of state x is $E(x) = -\frac{1}{2}xWx' + \theta x'$

Theorem:

Hopfield network converges to local minimum of energy function after a finite number of updates. \square

Proof: assume that x_k has been updated $\Rightarrow \tilde{x}_k = -x_k$ and $\tilde{x}_i = x_i$ for $i \neq k$

$$\begin{aligned} E(x) - E(\tilde{x}) &= -\frac{1}{2}xWx' + \theta x' + \frac{1}{2}\tilde{x}W\tilde{x}' - \theta \tilde{x}' \\ &= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} x_i x_j + \sum_{i=1}^n \theta_i x_i + \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} \tilde{x}_i \tilde{x}_j - \sum_{i=1}^n \theta_i \tilde{x}_i \\ &= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} (x_i x_j - \tilde{x}_i \tilde{x}_j) + \sum_{i=1}^n \theta_i \underbrace{(x_i - \tilde{x}_i)}_{=0 \text{ if } i \neq k} \\ &= -\frac{1}{2} \sum_{\substack{i=1 \\ i \neq k}}^n \sum_{j=1}^n w_{ij} \underbrace{(x_i x_j - \tilde{x}_i \tilde{x}_j)}_{\substack{\parallel \\ x_i}} - \frac{1}{2} \sum_{j=1}^n w_{kj} \underbrace{(x_k x_j - \tilde{x}_k \tilde{x}_j)}_{\substack{\parallel \\ 0 \text{ if } j = k \\ x_j \text{ if } j \neq k}} + \theta_k (x_k - \tilde{x}_k) \end{aligned}$$

$$= -\frac{1}{2} \sum_{\substack{i=1 \\ i \neq k}}^n \sum_{j=1}^n w_{ij} x_i \underbrace{(x_j - \tilde{x}_j)}_{=0 \text{ if } j \neq k} - \frac{1}{2} \sum_{\substack{j=1 \\ j \neq k}}^n w_{kj} x_j (x_k - \tilde{x}_k) + \theta_k (x_k - \tilde{x}_k)$$

$$= -\frac{1}{2} \sum_{\substack{i=1 \\ i \neq k}}^n w_{ik} x_i (x_k - \tilde{x}_k) - \frac{1}{2} \sum_{\substack{j=1 \\ j \neq k}}^n w_{kj} x_j (x_k - \tilde{x}_k) + \theta_k (x_k - \tilde{x}_k) \quad (\text{rename } j \text{ to } i, \text{ recall } W = W', w_{kk} = 0)$$

$$= -\sum_{i=1}^n w_{ik} x_i (x_k - \tilde{x}_k) + \theta_k (x_k - \tilde{x}_k)$$

$$= -(x_k - \tilde{x}_k) \left[\underbrace{\sum_{i=1}^n w_{ik} x_i}_{\text{excitation } e_k} - \theta_k \right] > 0 \quad \text{since:}$$

x_k	$x_k - \tilde{x}_k$	$e_k - \theta_k$	ΔE
+1	> 0	< 0	> 0
-1	< 0	> 0	> 0

> 0 if $x_k < 0$ and vice versa **q.e.d.**

Application to Combinatorial OptimizationIdea:

- transform combinatorial optimization problem as objective function with $x \in \{-1, +1\}^n$
- rearrange objective function to look like a Hopfield energy function
- extract weights W and thresholds θ from this energy function
- initialize a Hopfield net with these parameters W and θ
- run the Hopfield net until reaching stable state (= local minimizer of energy function)
- stable state is local minimizer of combinatorial optimization problem

Example I: Linear Functions

$$f(x) = \sum_{i=1}^n c_i x_i \rightarrow \min! \quad (x_i \in \{-1, +1\})$$

Evidently: $E(x) = f(x)$ with $W = 0$ and $\theta = c$

⇓

choose $x^{(0)} \in \{-1, +1\}^n$
set iteration counter $t = 0$

repeat

choose index k at random

$$x_k^{(t+1)} = \text{sgn}(x^{(t)} \cdot W_{\cdot,k} - \theta_k) = \text{sgn}(x^{(t)} \cdot 0 - c_k) = -\text{sgn}(c_k) = \begin{cases} -1 & \text{if } c_k > 0 \\ +1 & \text{if } c_k < 0 \end{cases}$$

increment t

until reaching fixed point

⇒ fixed point reached after $\Theta(n \log n)$ iterations on average

Example II: MAXCUT

given: graph with n nodes and symmetric weights $\omega_{ij} = \omega_{ji}$, $\omega_{ii} = 0$, on edges

task: find a partition $V = (V_0, V_1)$ of the nodes such that the weighted sum of edges with one endpoint in V_0 and one endpoint in V_1 becomes maximal

encoding: $\forall i=1, \dots, n: \quad y_i = 0 \Leftrightarrow \text{node } i \text{ in set } V_0; \quad y_i = 1 \Leftrightarrow \text{node } i \text{ in set } V_1$

objective function: $f(y) = \sum_{i=1}^{n-1} \sum_{j=i+1}^n \omega_{ij} [y_i(1-y_j) + y_j(1-y_i)] \rightarrow \max!$

preparations for applying Hopfield network

step 1: conversion to minimization problem

step 2: transformation of variables

step 3: transformation to "Hopfield normal form"

step 4: extract coefficients as weights and thresholds of Hopfield net

Example II: MAXCUT (continued)

step 1: conversion to minimization problem

$$\Rightarrow \text{multiply function with } -1 \Rightarrow E(y) = -f(y) \rightarrow \min!$$

step 2: transformation of variables

$$\Rightarrow y_i = (x_i + 1) / 2$$

$$\Rightarrow f(x) = \sum_{i=1}^{n-1} \sum_{j=i+1}^n \omega_{ij} \left[\frac{x_i + 1}{2} \left(1 - \frac{x_j + 1}{2} \right) + \frac{x_j + 1}{2} \left(1 - \frac{x_i + 1}{2} \right) \right]$$

$$= \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^n \omega_{ij} [1 - x_i x_j]$$

$$= \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^n \omega_{ij} - \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^n \omega_{ij} x_i x_j$$

constant value (does not affect location of optimal solution)

Example II: MAXCUT (continued)

step 3: transformation to "Hopfield normal form"

$$\begin{aligned} E(x) &= \frac{1}{2} \sum_{i=1}^{n-1} \sum_{j=i+1}^n \omega_{ij} x_i x_j = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \underbrace{\left(-\frac{1}{2} \omega_{ij} \right)}_{w_{ij}} x_i x_j \\ &= -\frac{1}{2} x' W x + \theta' x \\ &\quad \downarrow \\ &\quad 0' \end{aligned}$$

step 4: extract coefficients as weights and thresholds of Hopfield net

$$w_{ij} = -\frac{\omega_{ij}}{2} \text{ for } i \neq j, \quad w_{ii} = 0, \quad \theta_i = 0$$

remark: ω_{ij} : weights in graph — w_{ij} : weights in Hopfield net