第 8 章　人工神经网络及其应用

第8章 人工神经网络及其应用

- 人工神经网络是对人脑或生物神经网络若干基本特性的抽象和模拟。为机器学习等许多问题的研究提供了一条新的思路，目前已经在模式识别、机器视觉、联想记忆、自动控制、信号处理、软测量、决策分析、智能计算、组合优化问题求解、数据挖掘等方面获得成功应用。
- 本章着重介绍最基本、最典型、应用最广泛的**BP**神经网络和**Hopfield**神经网络及其在模式识别、联想记忆、软测量、智能计算、组合优化问题求解等方面的应用。

第8章 人工神经网络及其应用

- 神经网络（neural networks，NN）

- 生物神经网络(natural neural network, NNN): 由中枢神经系统（脑和脊髓）及周围神经系统（感觉神经、运动神经等）所构成的错综复杂的神经网络，其中最重要的是脑神经系统。

- 人工神经网络(artificial neural networks, ANN): 模拟人脑神经系统的结构和功能，运用大量简单处理单元经广泛连接而组成的人工网络系统。

神经网络方法： 隐式的
知识表示方法

第8章 人工神经网络及其应用

- **8.1 神经元与神经网络**
- **8.2 BP神经网络及其学习算法**
- **8.3 BP神经网络的应用**
- **8.4 Hopfield神经网络及其改进**
- **8.5 Hopfield神经网络的应用**

第8章 人工神经网络及其应用

- ✓ **8.1 神经元与神经网络**
- □ **8.2 BP神经网络及其学习算法**
- □ **8.3 BP神经网络的应用**
- □ **8.4 Hopfield神经网络及其改进**
- □ **8.5 Hopfield神经网络的应用**

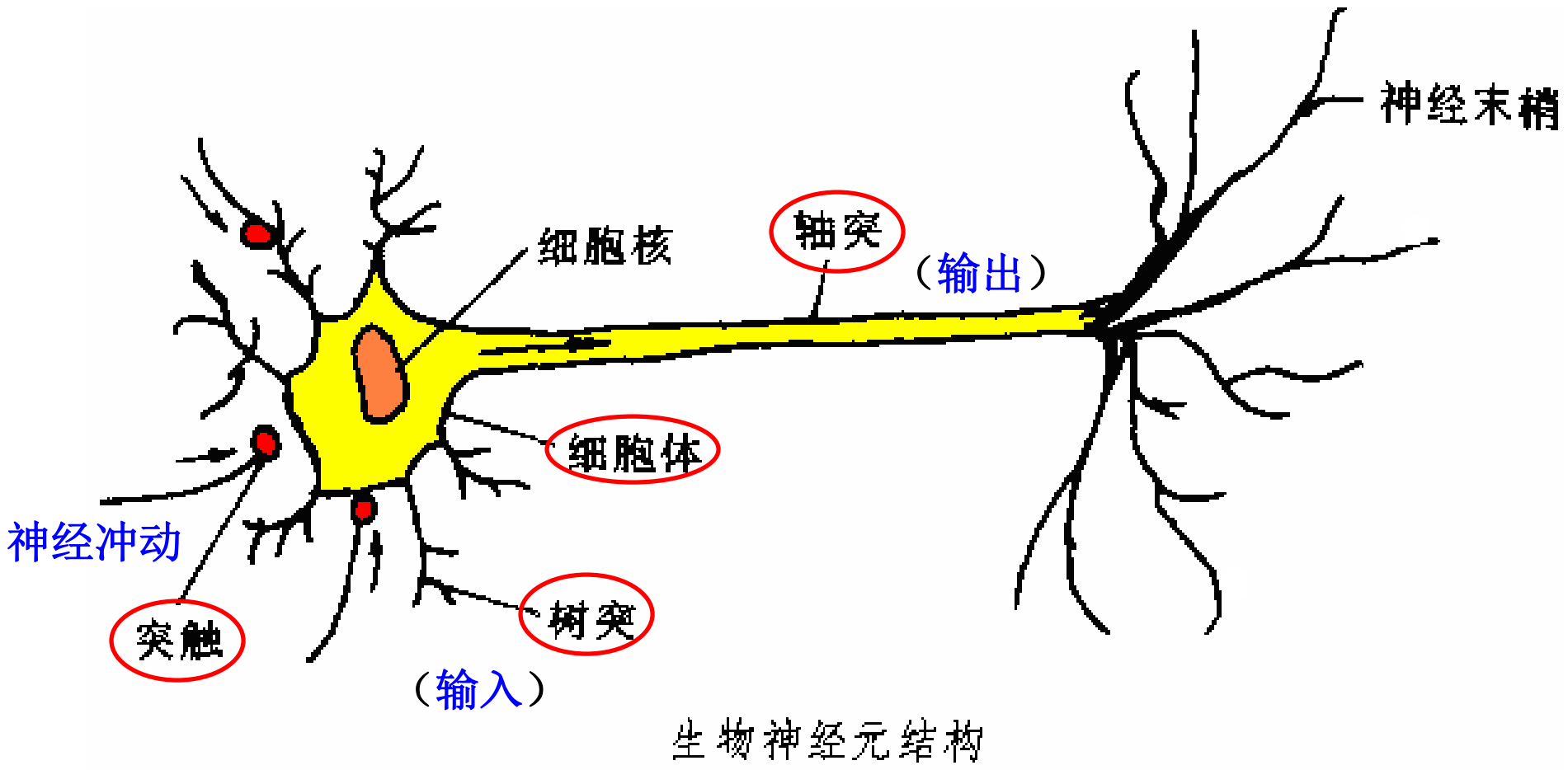
8.1 神经元与神经网络

- **8.1.1 生物神经元的结构**
- **8.1.2 神经元数学模型**
- **8.1.3 神经网络结构与工作方式**

8.1.1 生物神经元的结构

- 人脑由一千多亿（1011亿－ 1014 亿）个神经细胞（神经元）交织在一起的网状结构组成，其中大脑皮层约140亿个神经元，小脑皮层约1000亿个神经元。
- 神经元约有1000种类型，每个神经元大约与10^3— 10^4个其他神经元相连接，形成极为错综复杂而又灵活多变的神经网络。
- 人的智能行为就是由如此高度复杂的组织产生的。浩瀚的宇宙中，也许只有包含数千忆颗星球的银河系的复杂性能够与大脑相比。

8.1.1 生物神经元的结构



生物神经元结构

8.1.1 生物神经元的结构

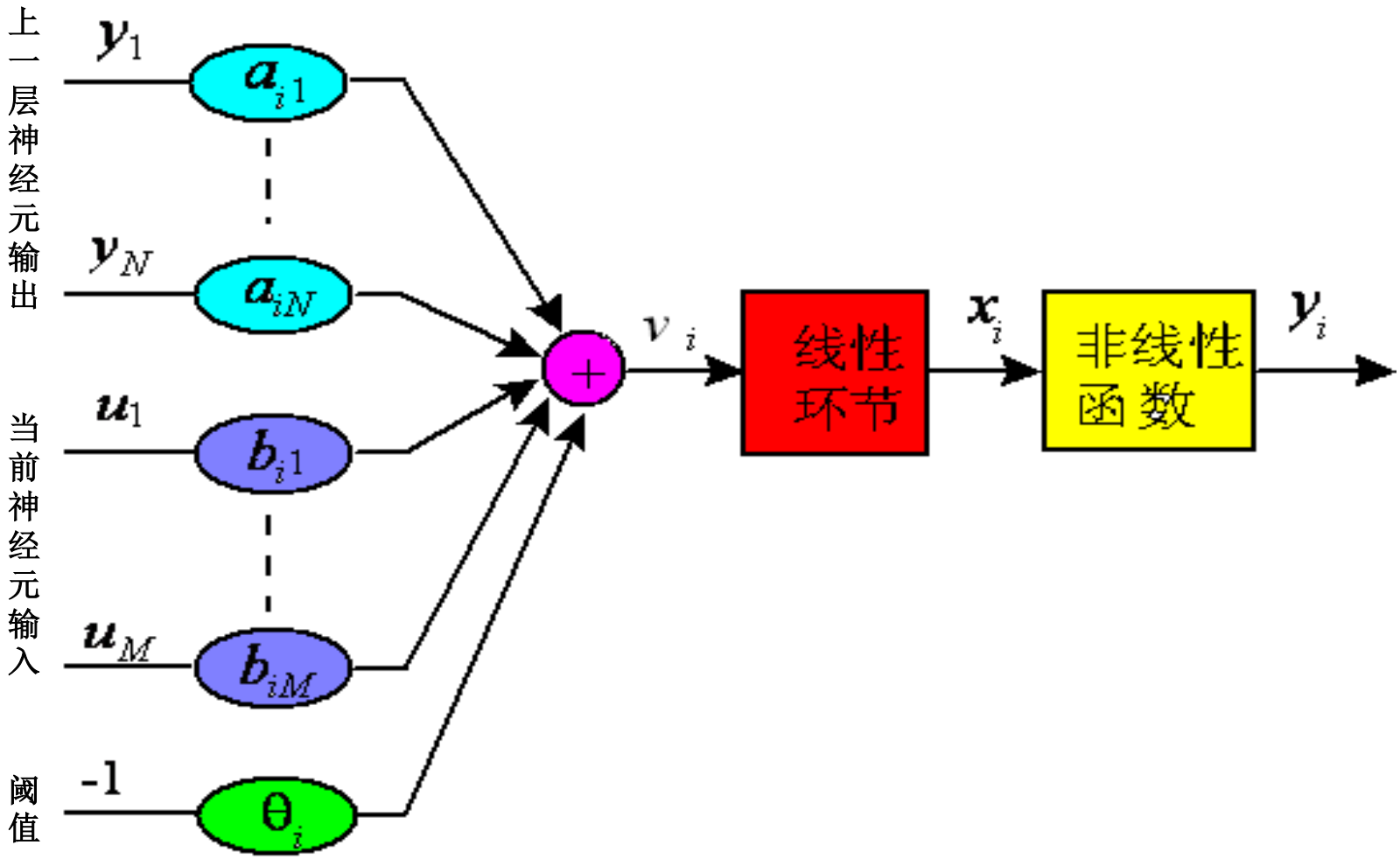
- 工作状态：
- 兴奋状态：细胞膜电位 > 动作电位的阈值 → 神经冲动
- 抑制状态：细胞膜电位 < 动作电位的阈值
- 学习与遗忘：由于神经元结构的可塑性，突触的传递作用可增强和减弱 。

8.1 神经元与神经网络

- **8.1.1 生物神经元的结构**
- **8.1.2 神经元数学模型**
- **8.1.3 神经网络的结构与工作方式**

8.1.2 神经元数学模型

1943年，麦克洛奇和皮兹提出M－P模型。一般模型：



8.1.2 神经元数学模型

$y_j(t)$：第j个神经元的输出。　　θ_i：第i个神经元的阈值。

$u_k(t)(k=1,2,\cdots,M)$：外部输入。　　a_{ij}, b_{ik}：权值。

- 加权求和：

$$v_i(t)=\sum_{\substack{j=1\\ j\to i}}^{N} a_{ij}y_j(t)+\sum_{k=1}^{M} b_{ik}u_k(t)-\theta_i$$

其矩阵形式：

$$V(t)=AY(t)+BU(t)-\theta$$

$$A=\{a_{ij}\}_{N\times N}\qquad B=\{b_{ik}\}_{N\times M}\qquad V=[v_1\ldots v_N]^T$$

$$U=[u_1\ldots u_M]^T\qquad \theta=[\theta_1\ldots\theta_N]^T\qquad Y=[y_1\ldots y_N]^T$$

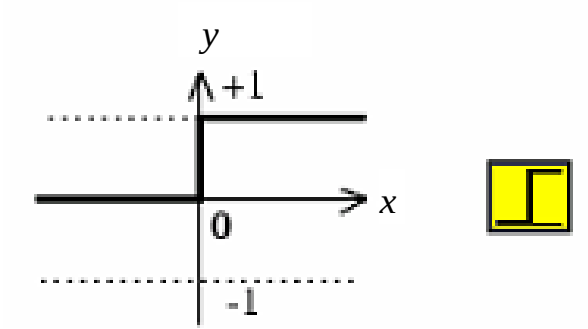
8.1.2 神经元数学模型

- 线性环节的传递函数：$X_i(s) = H(s)V_i(s)$

$H(s)$：**1**；$\frac{1}{s}$；$\frac{1}{Ts+1}$；e^{-Ts} 及其组合等。

8.1.2 神经元数学模型

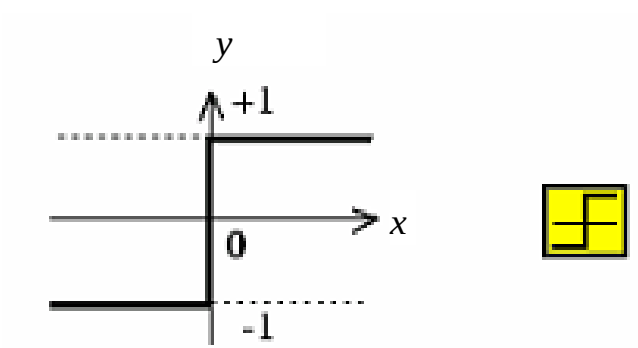
- 非线性激励函数（传输函数、输出变换函数）



$$y = hardlim(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$$

Hard-Limit Transfer Function

（硬极限函数或阶跃函数）



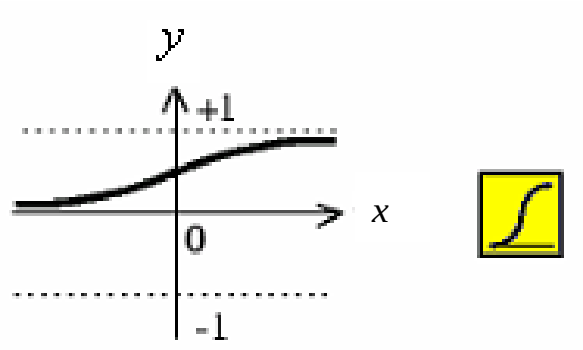
$$y = hardlims(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$

Symmetric Hard-Limit Trans. Funct.

（对称硬极限函数）

8.1.2 神经元数学模型

- 非线性激励函数（传输函数、输出变换函数）

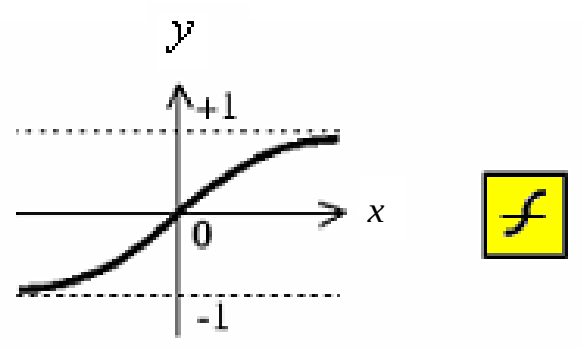


$$y = logsig(x) = \frac{1}{1 + e^{-\alpha x}}$$

$$\alpha = 1$$

Log-Sigmoid Transfer Function

（对数- S 形函数或S型函数）



$$y = tansig(x) = \frac{e^{\alpha x} - e^{-\alpha x}}{e^{\alpha x} + e^{-\alpha x}}$$

$$\alpha = 1$$

Tan-Sigmoid Transfer Function

（双曲正切S形函数）

8.1 神经元与神经网络

- **8.1.1 生物神经元的结构**
- **8.1.2 神经元的数学模型**
- **8.1.3 神经网络的结构与工作方式**

8.1.3 神经网络的结构与工作方式

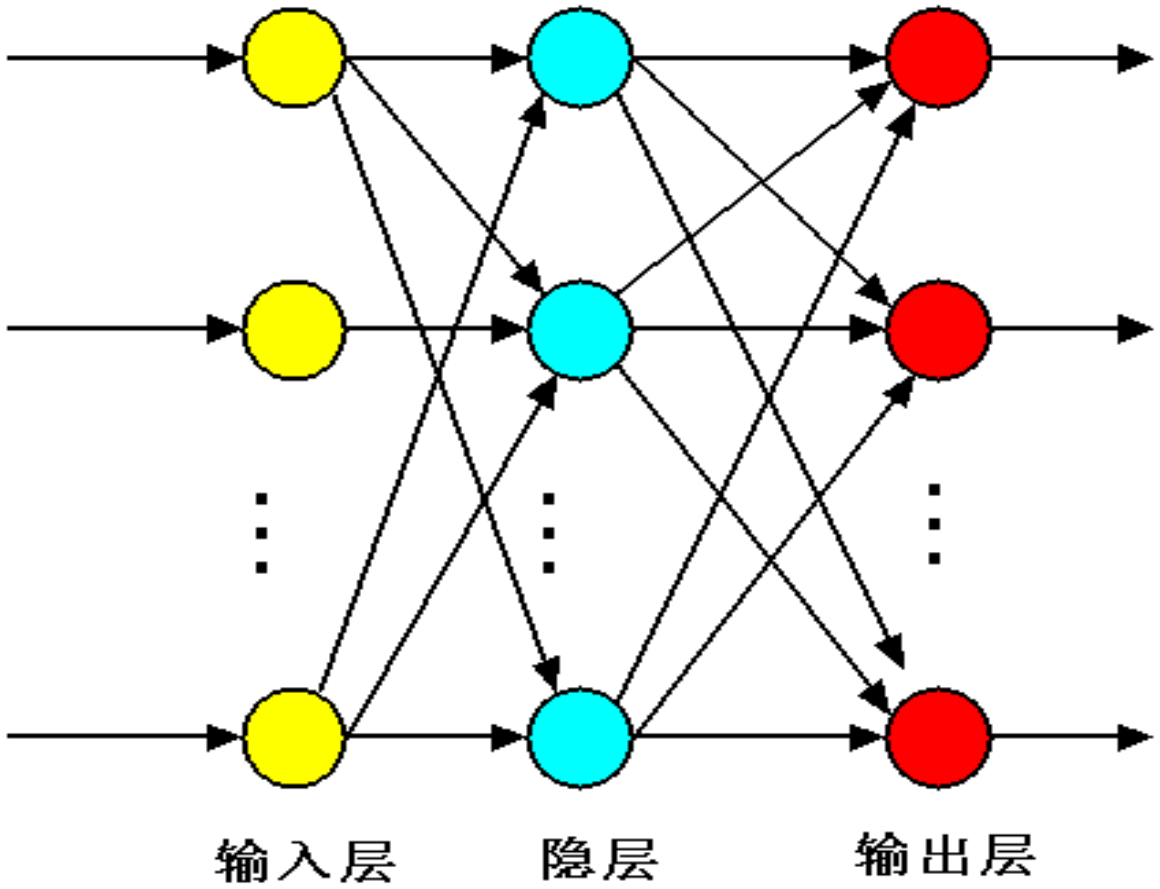
- 决定人工神经网络性能的三大要素：

  - 神经元的特性。
  - 神经元之间相互连接的形式——拓扑结构。
  - 为适应环境而改善性能的学习规则。

8.1.3 神经网络的结构与工作方式

1. 神经网络的结构

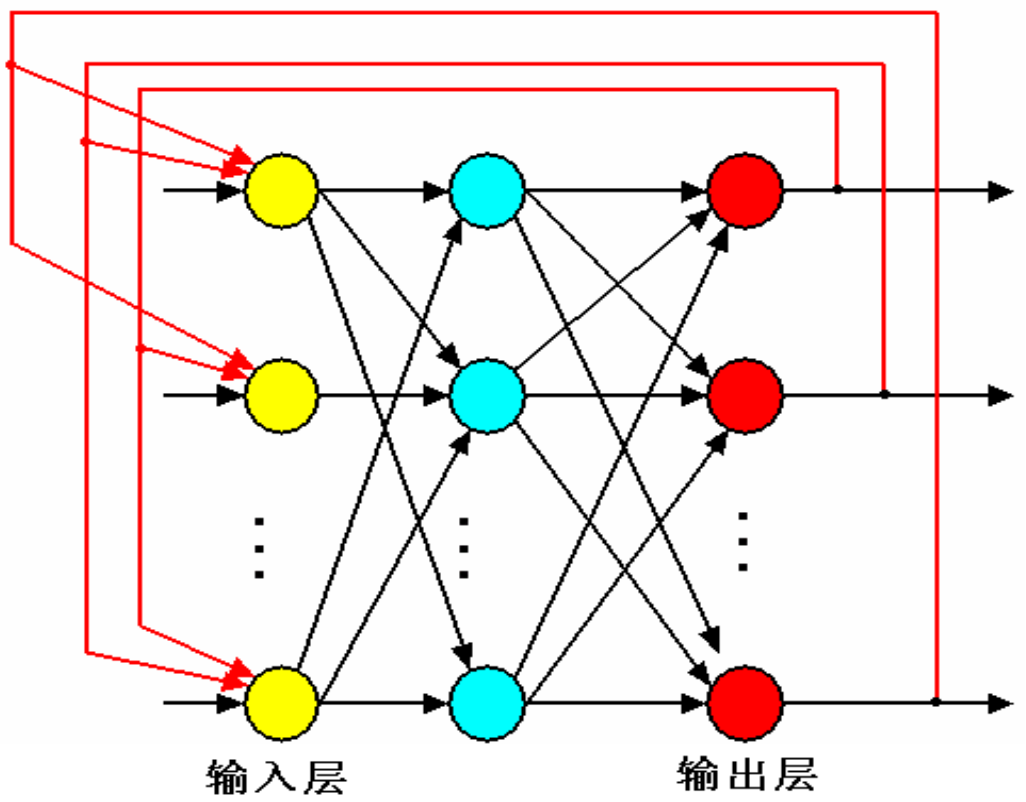
（1）前馈型（ 前向型）

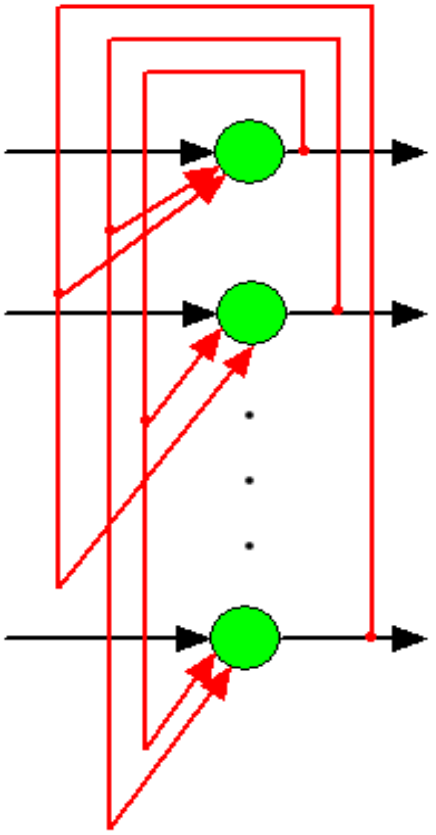


8.1.3 神经网络的结构与工作方式

❑ 1. 神经网络的结构

（2）反馈型





（Hopfield神经网络）
全互联反馈神经网络

8.1.3 神经网络的结构与工作方式

❑ 2. 神经网络的工作方式

- 同步（并行）方式：任一时刻神经网络中所有神经元同时调整状态。
- 异步（串行）方式：任一时刻只有一个神经元调整状态，而其它神经元的状态保持不变。

第8章 人工神经网络及其应用

- 8.1 神经元与神经网络
- ✓ 8.2 BP神经网络及其学习算法
- 8.3 BP神经网络的应用
- 8.4 Hopfield神经网络及其改进
- 8.5 Hopfield神经网络的应用

8.2 BP(Back Propagation)反向传播神经网络及其学习算法

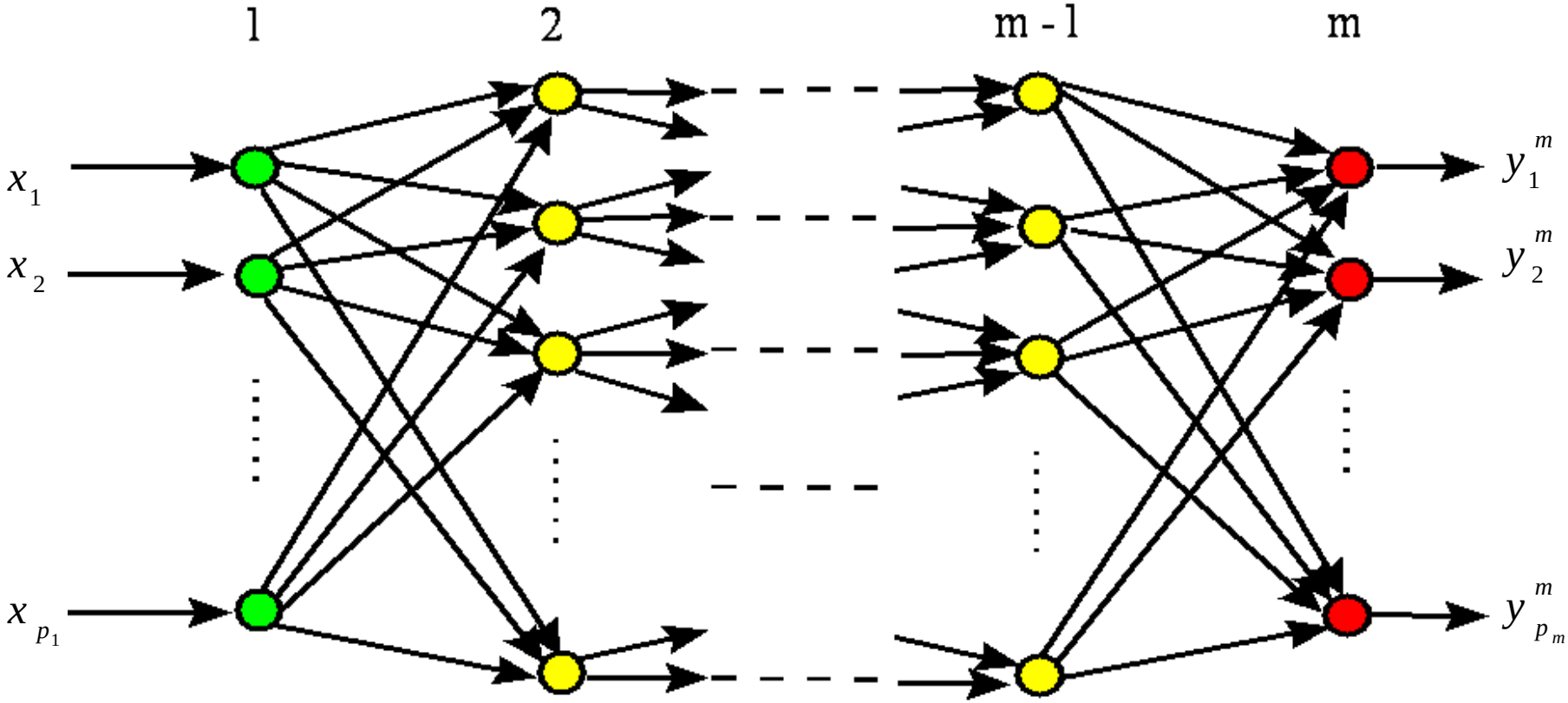
- **8.2.1 BP神经网络 的结构**
- **8.2.2 BP学习算法**
- **8.2.3 BP算法的实现**

8.2 BP神经网络及其学习算法

- **8.2.1 BP神经网络的结构**
- **8.2.2 BP学习算法**
- **8.2.3 BP算法的实现**

8.2.1 BP神经网络的结构

1. BP 网络结构



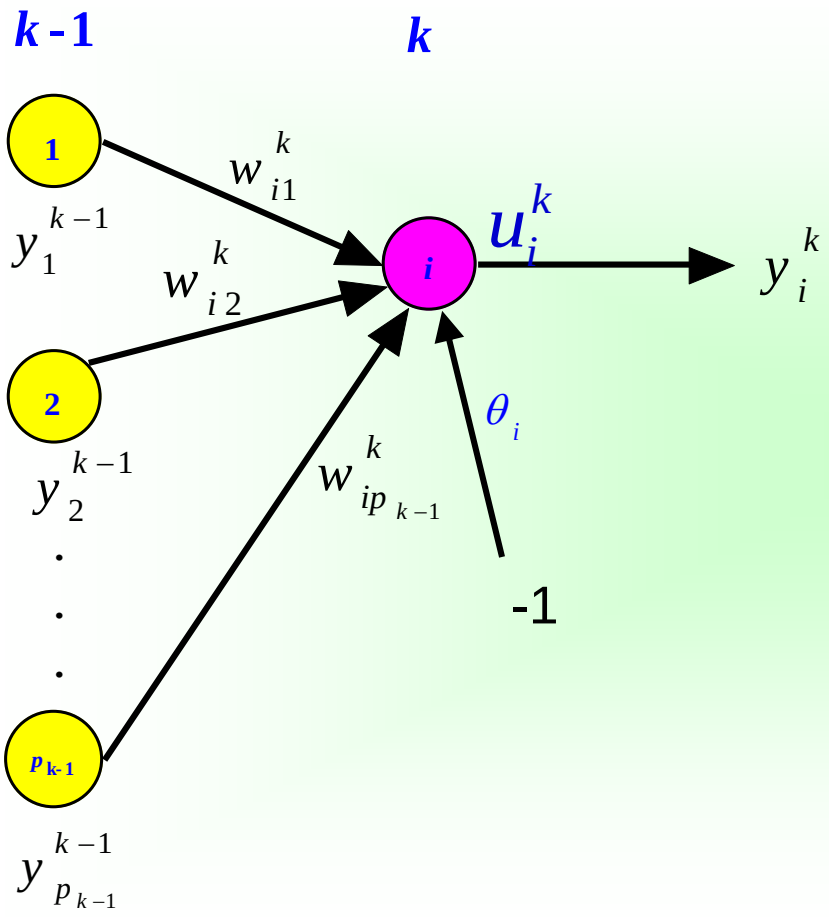
$$X = [x_1\ x_2 \ldots x_{p_1}]^T$$

$$Y = [y_1^m\ y_2^m \ldots y_{p_m}^m]^T$$

过程分为两个阶段：第一阶段是信号的前向传播，从输入层经过隐含层，最后到达输出层；第二阶段是误差的反向传播，从输出层到隐含层，最后到输入层，依次调节隐含层到输出层的权重和偏置，输入层到隐含层的权重和偏置。

8.2.1 BP神经网络的结构

2. 输入输出变换关系



$$u_i^k = \sum_{\substack{j=1 \\ j\to i}}^{p_{k-1}} w_{ij}^k y_j^{k-1} - \theta_i = \sum_{j=0}^{p_{k-1}} w_{ij}^k y_j^{k-1}$$

$$(y_0^{k-1} = \theta_i, w_{i0}^k = -1)$$

$$y_i^k = f(u_i^k) = \frac{1}{1+e^{-u_i^k}}$$

$k = 1,2,\ldots, m$ 第k层

$i = 1,2,\ldots, p_k$ 第k层的第i个神经元

8.2.1 BP神经网络的结构

3. 工作过程

- 第一阶段或网络训练阶段：
  - N 组输入输出样本：$x_{d1}=[x_{d1}, x_{d2},\ldots, x_{dp1}]^T$

$$y_d=[y_{d1}, y_{d2},\cdots,y_{dpm}]^T$$

$$d=1, 2,\ldots, N$$

  - 对网络的连接权进行学习和调整，以使该网络实现给定样本的输入输出映射关系。
- 第二阶段或称工作阶段：把实验数据或实际数据输入到网络，网络在误差范围内预测计算出结果。

8.2 BP神经网络及其学习算法

- **8.2.1 BP神经网络的结构**
- **8.2.2 BP学习算法**
- **8.2.3 BP算法的实现**

8.2.2 BP学习算法

- 两个问题：

- （1）是否存在一个BP神经网络能够逼近给定的样本或者函数。

> Kolmogorov 定理：给定任意 $\varepsilon > 0$，对于任意的 L_2 型连续函数 $f:[0,1]^n \longrightarrow R^m$，存在一个三层 BP 神经网络，其输入层有 n 个神经元，中间层有 $2n+1$ 个神经元，输出层有 m 个神经元，它可以在任意 ε 平方误差精度内逼近 f。

- （2）如何调整BP神经网络的连接权，使网络的输入与输出与给定的样本相同。
  - 1986年，鲁梅尔哈特（D. Rumelhart）等提出BP学习算法。

8.2.2 BP学习算法

1. 基本思想

- 目标函数：

$$J = \frac{1}{2}\sum_{i=1}^{p_m}(y_i^m - y_{si})^2$$

- 约束条件：

$$u_i^k = \sum_{j,\, j\to i} w_{ij}^{k-1} y_j^{k-1}$$

$i = 1,2,\ldots,p_k$ 第k层的第i个神经元

$j \to i$: 第i个神经元的所有输入

$$y_i^k = f_k(u_i^k)$$

$k = 1,2,\ldots m$ 第k层

- 连接权值的修正量：

$$\Delta w_{ij}^{k-1} = -\varepsilon \frac{\partial J}{\partial w_{ij}^{k-1}} \qquad j = 1,2,\ldots,\ p_{k-1}$$

8.2.2 BP学习算法

先求 $$\frac{\partial J}{\partial w_{ij}^{k-1}}=\frac{\partial J}{\partial u_i^k}\frac{\partial u_i^k}{\partial w_{ij}^{k-1}}=\frac{\partial J}{\partial u_i^k}\frac{\partial}{\partial w_{ij}^{k-1}}\left(\sum_j w_{ij}^{k-1}y_j^{k-1}\right)=\frac{\partial J}{\partial u_i^k}y_j^{k-1}$$

记 $$d_i^k=\frac{\partial J}{\partial u_i^k}=\frac{\partial J}{\partial y_i^k}\frac{\partial y_i^k}{\partial u_i^k}\underline{\underline{y_i^k=f(u_i^k)}}\frac{\partial J}{\partial y_i^k}f_k'(u_i^k)$$

（1）对输出层的神经元 $$\frac{\partial J}{\partial y_i^k}\underline{\underline{k=m}}\frac{\partial J}{\partial y_i^m}\underline{\underline{J=\frac{1}{2}\sum_{i=1}^{p_m}(y_i^m-y_{si})^2}}\quad y_i^m-y_{si}$$

则 $$d_i^m=(y_i^m-y_{si})f_m'(u_i^m)$$

（2）对隐单元层，有 $$\frac{\partial J}{\partial y_i^k}=\sum_l\frac{\partial J}{\partial u_l^{k+1}}\frac{\partial u_l^{k+1}}{\partial y_i^k}=\sum_l d_l^{k+1}w_{li}^k$$

则 $$d_i^k=f_k'(u_i^k)\sum_l d_l^{k+1}w_{li}^k$$

8.2.2 BP学习算法

- **2. 学习算法**

$$\Delta w_{ij}^{k-1} = -\varepsilon d_i^k y_j^{k-1}$$

$$d_i^m = (y_i^m - y_{si}) f_m'(u_i^m)$$

——输出层连接权调整公式

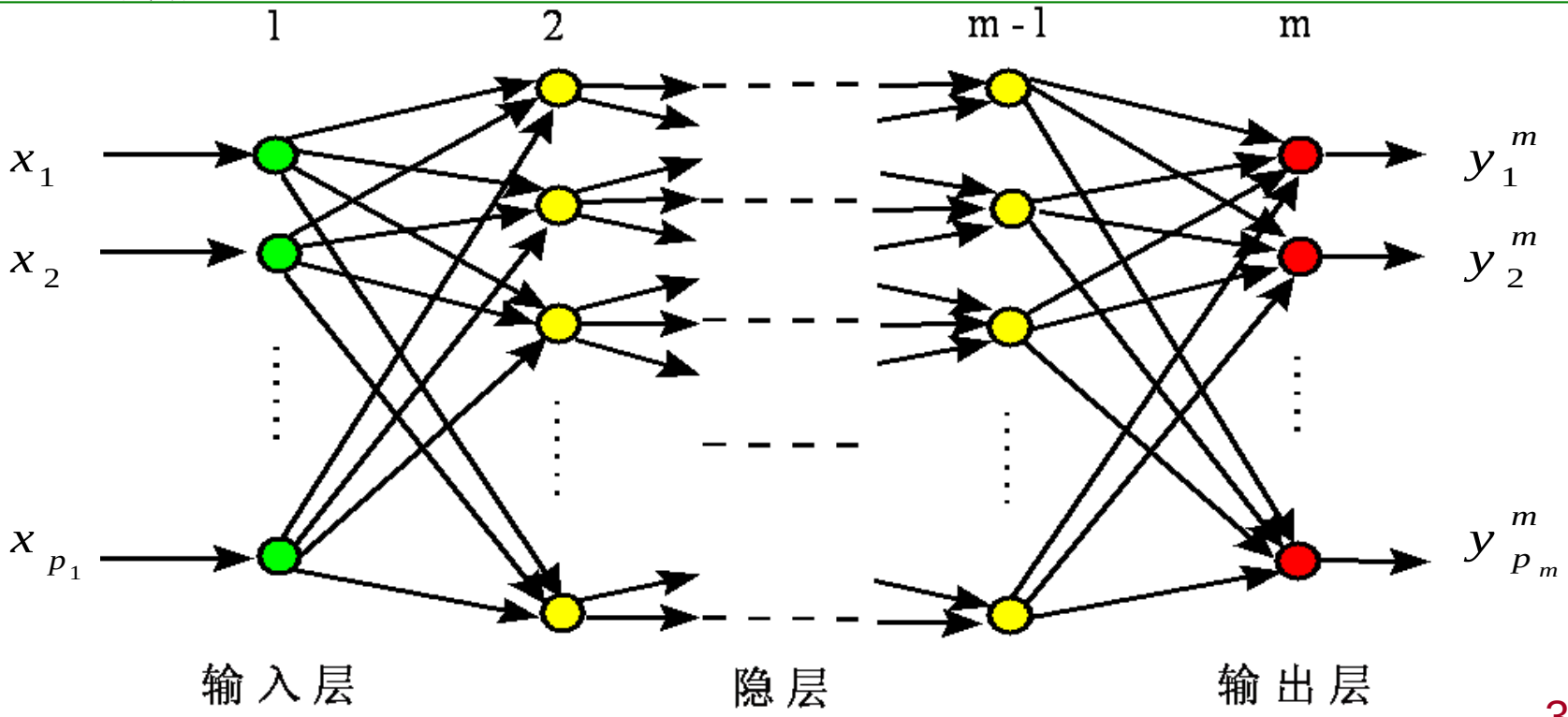
$$d_i^k = f_k'(u_i^k)\sum_l d_l^{k+1} w_{li}^k$$

——隐层连接权调整公式

8.2.2 BP学习算法

2. 学习算法

- 正向传播：输入信息由输入层传至隐层，最终在输出层输出。
- 反向传播：从输出层往前修改各层神经元的权值，使误差信号最小。
- BP神经网络是一种多层的前馈神经网络，主要特点：信号是前向传播，而误差是反向传播。



8.2.2 BP学习算法

- **2. 学习算法**

当 $y_i^k = f_k(u_i^k) = \dfrac{1}{1+e^{-u_i^k}}$ 时

$$\Delta w_{ij}^{k-1} = -\varepsilon d_i^k y_j^{k-1}$$

$$d_i^m = y_i^m (1 - y_i^m)(y_i^m - y_i)$$

——输出层连接权调整公式

$$d_i^k = y_i^k (1 - y_i^k) \sum_l w_{li}^k d_l^{k+1}$$

——隐层连接权调整公式

8.2 BP神经网络及其学习算法

- **8.2.1 BP神经网络的结构**
- **8.2.2 BP学习算法**
- **8.2.3 BP算法的实现**

8.2.3 BP算法的实现

- **1. BP算法的设计**

（1） 隐层数及隐层神经元数的确定：目前尚无理论指导。

（2）初始权值的设置：一般以一个均值为0的随机分布设置网络的初始权值。

（3）训练数据预处理：线性的特征比例变换，将所有的特征变换到[0，1]或者[-1，1]区间内，使得在每个训练集上，每个特征的均值为0，并且具有相同的方差。

（4）后处理过程：当应用神经网络进行分类操作时，通常将输出值编码成所谓的名义变量，具体的值对应类别标号。

8.2.3 BP算法的实现

- **2. BP算法的计算机实现流程**

（1）初始化：对所有连接权和阈值赋以随机任意小值；

$$w_{ij}^{k}(t),\quad \theta_{i}^{k}(t),(k=1,\ldots,\ m;i=1,\ldots,\ p_{k};\ j=1,\ldots,\ p_{k-1};t=0)$$

（2） 从 *N* 组输入输出样本中取一组样本：$x=[x_1, x_2,\ldots, x_{p1}]^T$，$y=[y_1, y_2,\ldots,y_{pm}]^T$，把输入信息$x=[x_1, x_2,\ldots, x_{p1}]^T$输入到BP网络中

（3）正向传播：计算各层节点的输出：

$$y_{i}^{k}\qquad (i=1,\ldots,\ p_{k};\quad k=1,\ldots,\ m)$$

（4）计算网络的实际输出与期望输出的误差：

$$e_{i}=y_{i}-y_{i}^{m}\qquad (i=1,\ldots,\ p_{m})$$

8.2.3 BP算法的实现

- **2. BP算法的计算机实现流程**

（5）反向传播：从输出层方向计算到第一个隐层，按连接权值修正公式向减小误差方向调整网络的各个连接权值。

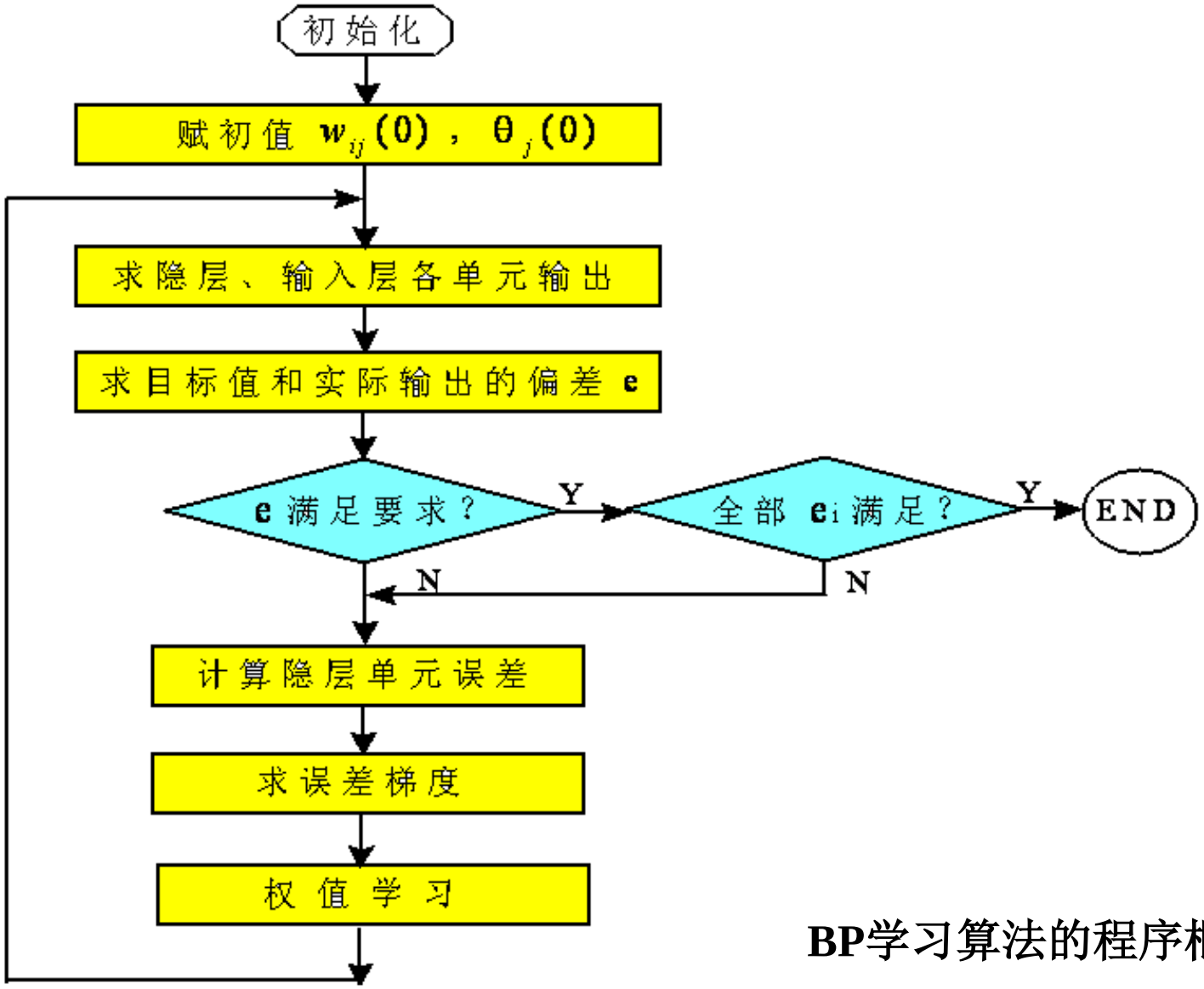
$$\Delta w_{ij}^{k-1} = -\varepsilon d_i^k y_j^{k-1}$$

$$d_i^m = y_i^m (1 - y_i^m)(y_i^m - y_i) \quad \text{——输出层连接权调整公式}$$

$$d_i^k = y_i^k (1 - y_i^k) \sum_{l=1}^{p_{k+1}} w_{li}^{k+1} d_l^{k+1} \quad \text{——隐层连接权调整公式}$$

（6）让$t+1 \rightarrow t$，取出另一组样本重复（2）—（5），直到 N 组输入输出样本的误差达到要求时为止。

8.2.3 BP算法的实现



BP学习算法的程序框图

8.2.4 BP算法的特点分析

1. 特点

- **BP网络：多层前向网络（输入层、隐层、输出层）。**
- **连接权值：通过Delta学习算法进行修正。**
- **神经元传输函数：S形函数。**
- **学习算法：正向传播、反向传播。**
- **层与层的连接是单向的，信息的传播是双向的。**

8.2.4 BP算法的特点分析

2. BP网络的主要优缺点

- 优点

- 很好的逼近特性。
- 具有较强的泛化能力。
- 具有较好的容错性。

- 缺点

- 收敛速度慢。
- 局部极值。
- 难以确定隐层和隐层结点的数目。

8.3 BP神经网络的应用

- **8.3.1 BP神经网络在模式识别中的应用**
- **8.3.2 BP神经网络在软测量中的应用**

8.3.1 BP神经网络在模式识别中的应用

模式识别研究用计算机模拟生物、人的感知，对模式信息，如图像、文字、语音等，进行识别和分类。

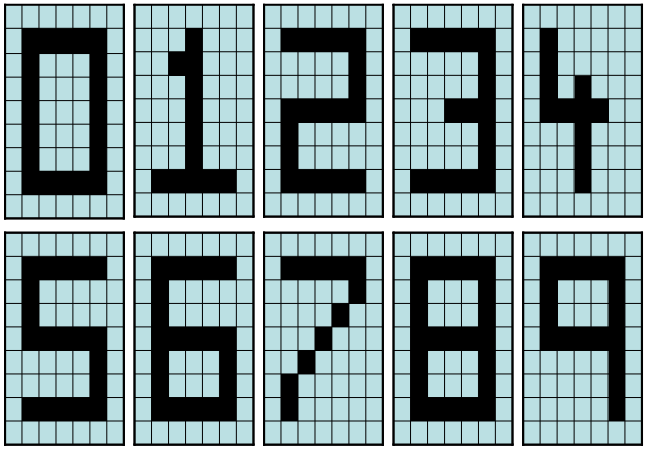
传统人工智能的研究部分地显示了人脑的归纳、推理等智能。但是，对于人类底层的智能，如视觉、听觉、触觉等方面，现代计算机系统的信息处理能力还不如一个幼儿园的孩子。

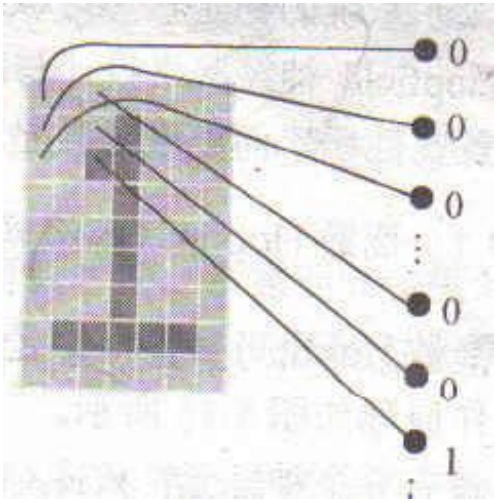
神经网络模型模拟了人脑神经系统的特点：处理单元的广泛连接；并行分布式信息储存、处理；自适应学习能力等。

神经网络模式识别方法具有较强的容错能力、自适应学习能力、并行信息处理能力。

8.3.1 BP神经网络在模式识别中的应用

例8.1 设计一个三层BP网络对数字0至9进行分类。





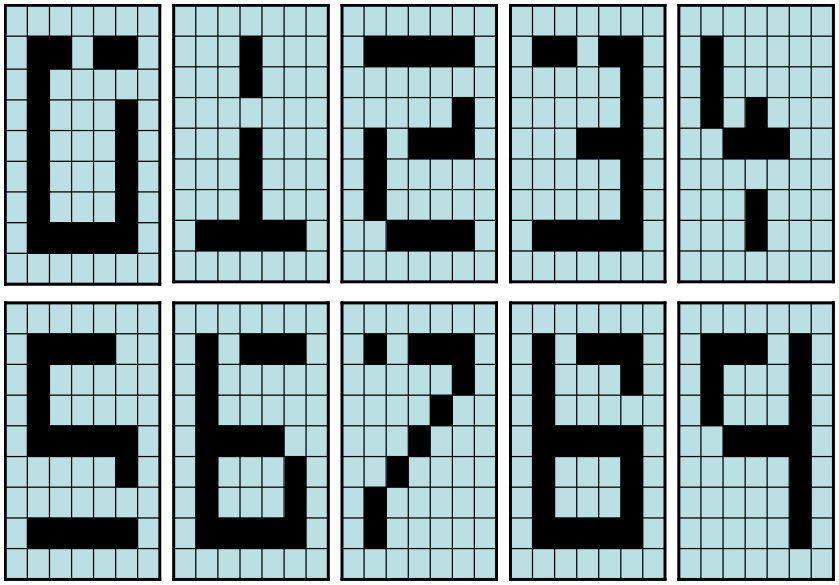
每个数字用9×7的网格表示，灰色像素代表0，黑色像素代表1。将每个网格表示为0，1的长位串。位映射由左上角开始向下直到网格的整个一列，然后重复其他列。

选择BP网络结构为63-6-9。97个输入结点，对应上述网格的映射。9个输出结点对应10种分类。

使用的学习步长为0.3。训练1000个周期，如果输出结点的值大于0.9，则取为ON，如果输出结点的值小于0.1，则取为OFF。

8.3.1 BP神经网络在模式识别中的应用

当训练成功后，对如图所示测试数据进行测试。测试数据都有一个或者多个位丢失。



测试结果表明：除了8以外，所有被测的数字都能够被正确地识别。

对于数字8，神经网络的第6个结点的输出值为0.53，第8个结点的输出值为0.41，表明第8个样本是模糊的，可能是数字6，也可能是数字8，但也不完全确信是两者之一。

第8章 人工神经网络及其应用

- 8.1 神经元与神经网络
- 8.2 BP神经网络及其学习算法
- 8.3 BP神经网络的应用
- ✓ **8.4 Hopfield神经网络及其改进**
- 8.5 Hopfield神经网络的应用

8.4 Hopfield神经网络及其改进

- **8.4.1 离散型Hopfield神经网络**
- **8.4.2 连续型Hopfield神经网络及其VLSI实现**
- **8.4.3 随机神经网络**

8.4 Hopfield神经网络及其改进

- **8.4.1 离散型Hopfield神经网络**
- **8.4.2 连续型Hopfield神经网络及其VLSI实现**
- **8.4.3 随机神经网络**

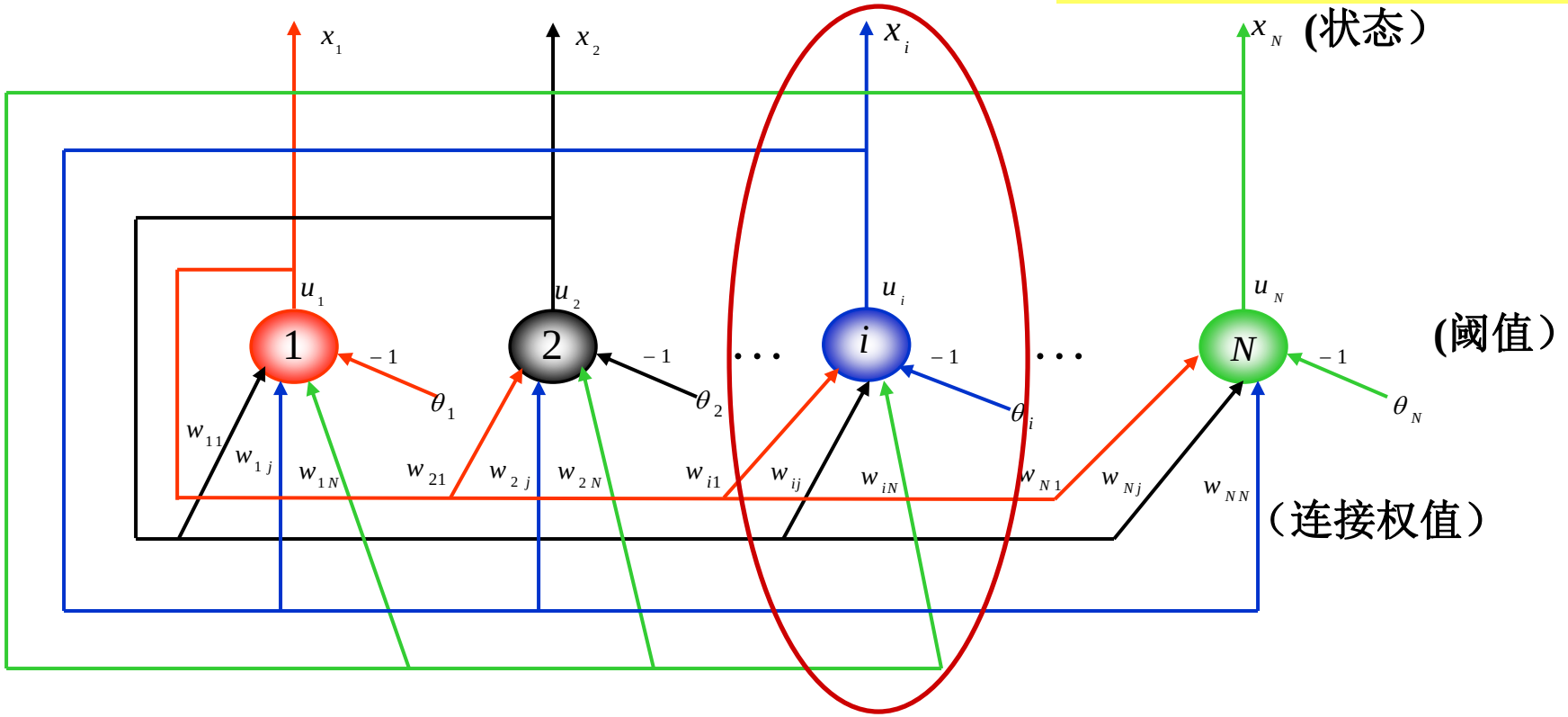
8.4.1 离散Hopfield 神经网络

1. 离散Hopfield神经网络模型

- 网络结构：

$$u_i(k+1) = \sum_{\substack{j=1 \\ j\neq i \\ j\to i}}^{N} w_{ij} x_j(k) - \theta_i$$

$$w_{ij} = 0 \quad i = j$$



离散Hopfield神经网络结构图

8.4.1 离散Hopfield 神经网络

1. 离散Hopfield神经

- 输入输出关系：

$$x(k+1) = f(W_x(k) - \theta), \forall i$$

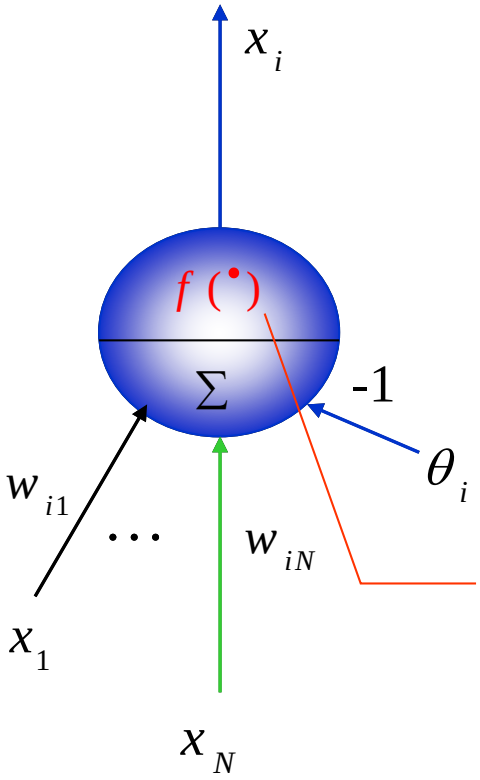
$$\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_N]^T \qquad \boldsymbol{\theta} = [\theta_1 \quad \theta_2 \quad \cdots \quad \theta_N]^T$$

$$\mathbf{W} = [w_{ij}]_{N\times N} \qquad \mathbf{f}(s) = [f(s_1) \quad f(s_2) \quad \cdots \quad f(s_N)]^T$$

$$x_i(k+1) = f\left(\sum_{\substack{j=1 \\ j\to i}}^{N} w_{ij}x_j(k) - \theta_i\right)$$

连接权重　阈值

注：$w_{ii}=0$



$$f(s) = \begin{cases} 1 & s \geq 0 \\ -1 & s < 0 \end{cases} \quad \text{或} \quad f(s) = \begin{cases} 1 & s \geq 0 \\ 0 & s < 0 \end{cases}$$

8.4.1 离散Hopfield 神经网络

1. 离散Hopfield神经网络模型

- 工作方式：

异步（串行）方式：

$$\begin{cases} x_i(k+1) = f\left(\sum_{j=1}^{N} w_{ij} x_j(k) - \theta_i\right) \\ x_j(k+1) = x_j(k), \qquad j \neq i \end{cases}$$

神经元一个一个改变状态

同步（并行）方式：$x_i(k+1) = f\left(\sum_{j=1}^{N} w_{ij} x_j(k) - \theta_i\right), \forall i$

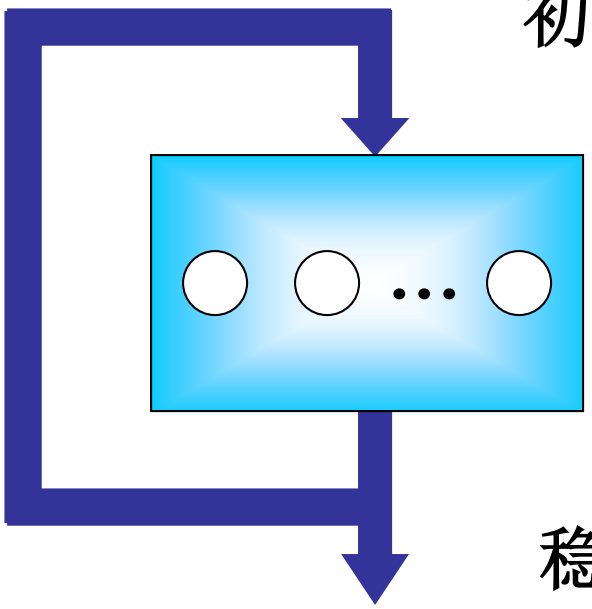
神经元在一轮中同时改变状态

$$x(k+1) = f(W_x(k) - \theta), \forall i$$

8.4.1 离散Hopfield 神经网络

1. 离散Hopfield神经网络模型

- 工作过程：



初态：$x(0)=[x_1(0),\quad x_2(0),\quad \cdots,\quad x_N(0)]^{\mathrm{T}}$

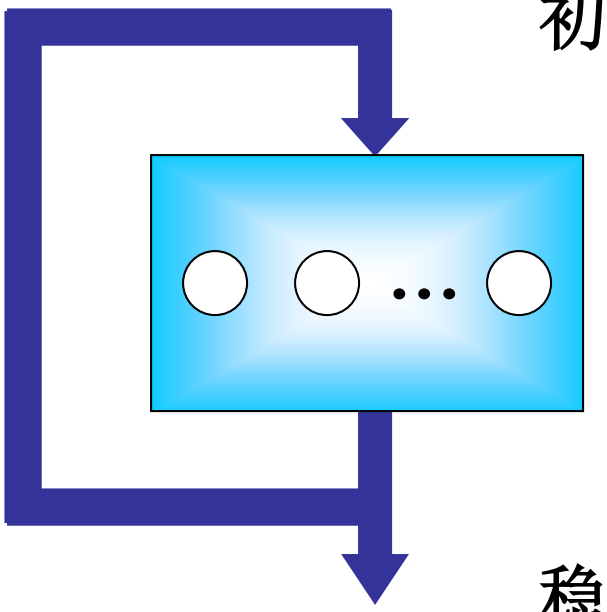
（异步或同步方式）

稳态：$\lim\limits_{k\to\infty} x(k)$

8.4.1 离散Hopfield 神经网络

1. 离散Hopfield神经网络模型

- 工作过程：



初态 ⟶ 记忆样本的部分信息

联想
记忆

稳态 ⟶ 记忆样本

8.4.1 离散Hopfield 神经网络

2. 网络的稳定性

- 稳定性定义：

- 若从某一时刻开始，网络中所有神经元的状态不再改变，即$x = f(w_x-\theta)$，则称该网络是稳定的，x为网络的稳定点或吸引子 。

$$x(k) = f(W_x(k) - \theta) = x(k+1)$$

- Hopfield神经网络是高维非线性系统，可能有许多稳定优态。从任何初始状态开始运动，总可以到某个稳定状态。这些稳定状态可以通过改变网络参数得到。

8.4.1 离散Hopfield 神经网络

2. 网络的稳定性

- 稳定性定理证明：1983年，科恩（Cohen）、葛劳斯伯格（S. Grossberg）。
- 稳定性定理（Hopfield）
  - 串行稳定性 —— W：对称阵
  - 并行稳定性 —— W：非负定对称阵

8.4 Hopfield神经网络及其改进

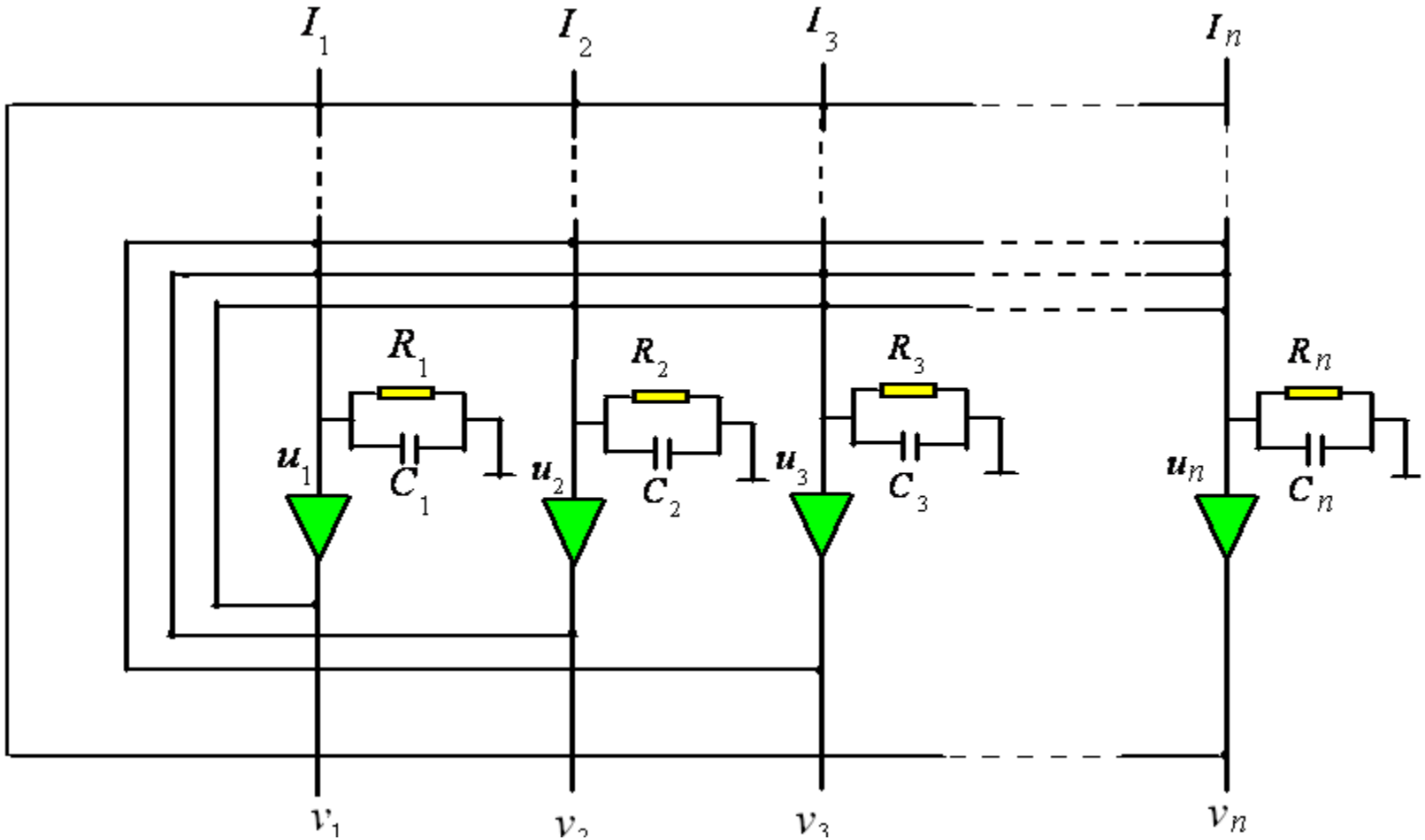
- **8.4.1 离散型Hopfield神经网络**
- **8.4.2 连续型Hopfield神经网络及其VLSI实现**

8.4.2 连续型Hopfield神经网络及其VLSI实现

- 连续**Hopfield**神经网络模型
- 网络的稳定性
- 应用举例

8.4.2 连续型Hopfield神经网络及其VLSI实现

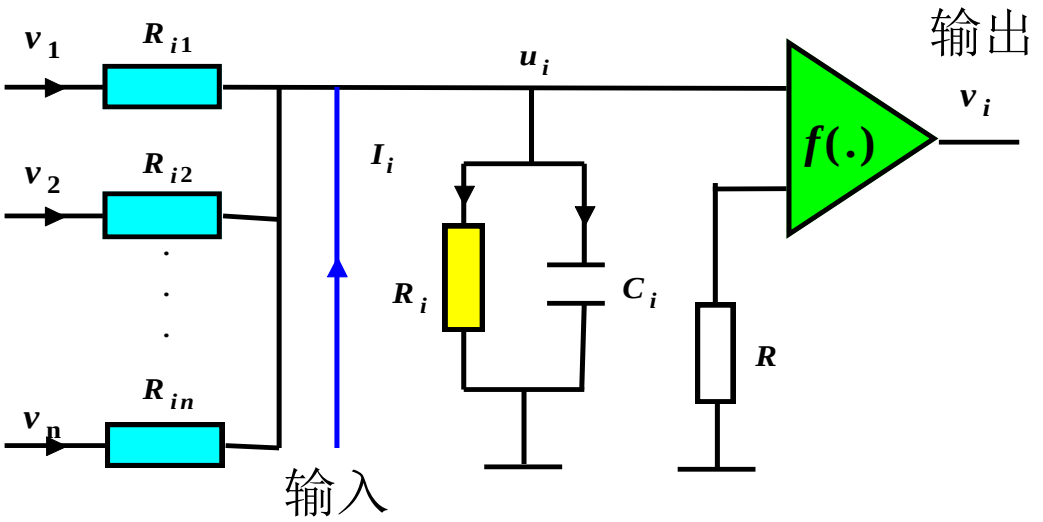
1. 连续Hopfield神经网络模型



利用电阻、电容和运算放大器等元件组成的模拟电路来实现对网络神经元的描述，把最优化问题的目标函数转换成Hopfield神经网络的能量函数，通过网络能量函数最小化来寻找对应问题的最优解

8.4.2 连续型Hopfield神经网络及其VLSI实现

1. 连续Hopfield神经网络模型



$$\begin{cases} \dfrac{u_i}{R_i} + C_i \dfrac{du_i}{dt} = \sum_{j=1}^{n} \dfrac{v_j - u_i}{R_{ij}} + I_i \\ v_i = f(u_i) \end{cases}$$

$$\frac{1}{R'_i} = \frac{1}{R_i} + \sum_{j=1}^{n} \frac{1}{R_{ij}}$$

$$w_{ij} = \frac{1}{R_{ij}}$$

$$C_i \frac{du_i}{dt} = -\frac{u_i}{R'_i} + \sum_{j=1}^{n} w_{ij} v_j + I_i$$

$$v_i = f(u_i) = \frac{1}{1 + e^{-\frac{2u_i}{u_0}}} = \frac{1}{2}[1 + \tanh(\frac{u_i}{u_0})]$$

8.4.2 连续型Hopfield神经网络及其VLSI实现

2. 网络的稳定性

- 计算能量函数：

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} v_i v_j - \sum_{i=1}^{N} v_i I_i + \sum_{i=1}^{N} \frac{1}{R'_i} \int_0^{v_i} f^{-1}(v)dv$$

- 定理：对于连续型 **Hopfield** 神经网络，若 $f^{-1}(\cdot)$ 为单调递增的连续函数，$C_i > 0$，$w_{ij} = w_{ji}$，则 $\frac{dE}{dt} \le 0$；当且仅当 $\frac{dv_i}{dt} = 0$，（$1 \le i \le N$）时，$\frac{dE}{dt} = 0$

8.4.3 随机神经网络

- **Hopfield神经网络中，神经元状态为1是根据其输入是否大于阈值确定的，是确定性的。**

- **随机神经网络中，神经元状态为1是随机的，服从一定的概率分布。例如，服从玻尔兹曼(Boltzmann)、高斯(Gaussian)、柯西(Cauchy)分布等，从而构成玻尔兹曼机、高斯机、柯西机等随机机。**

8.4.3 随机神经网络

1. Boltzmann机

- 1985年，加拿大多伦多大学教授欣顿(Hinton)等人借助统计物理学的概念和方法，提出了Boltzmann机神经网络模型。
- Boltzmann机是离散Hopfield神经网络的一种变型，通过对离散Hopfield神经网络加以扰动，使其以概率的形式表达，而网络的模型方程不变，只是输出值类似于Boltzmann分布以概率分布取值。
- Boltzmann机是按Boltzmann概率分布动作的神经网络。

8.4.3 随机神经网络

1. Boltzmann机 （续）

- 离散Hopfield神经网络的输出：

$$v_i(t+1) = \mathrm{sgn}\left(\sum_{\substack{j=1 \\ j\neq i}}^{n} w_{ij} v_j(t) - \theta_i\right)$$

- Boltzman机的内部状态：

$$I_i = \sum_{\substack{j=1 \\ j\neq i}}^{n} w_{ij} v_j(t) - \theta_i$$

- 神经元i输出值为0和1时的概率：

$$p_i(1) = \frac{1}{1+\mathrm{e}^{-I_i/T}} \quad p_i(0) = 1 - p_i(1)$$

T类似温度的扰动

8.4.3 随机神经网络

1. Boltzmann机 （续）

- Boltzmann的能量函数：

$$E = -\frac{1}{2}\sum_{i}\sum_{j\neq i} w_{ij} v_i v_j - \sum_{i} v_i \theta_i$$

- 神经元*i*状态转换时网络能量的变化：

$$\Delta E_i = \sum_{j} w_{ij} v_j + \theta_i$$

- 神经元*i*改变为状态“1”的概率：

$$p_i = \frac{1}{1+\exp(-\frac{\Delta E_i}{T})}$$

8.4.3 随机神经网络

2. 高斯机

$$\frac{du_i}{dt} = \sum_{j=1}^{n} w_{ij} v_j + I_i + \eta$$

η :均值为0的高斯随机变量（白噪声），其方差为 $\sigma = cT$

3. 柯西机

$$\frac{du_i}{dt} = \sum_{j=1}^{n} w_{ij} v_j + I_i + \eta$$

η ：柯西随机变量（有色噪声）

8.5 Hopfield神经网络的应用

- **8.5.1 Hopfield神经网络在联想记忆中的应用**
- **8.5.2 Hopfield神经网络优化方法**

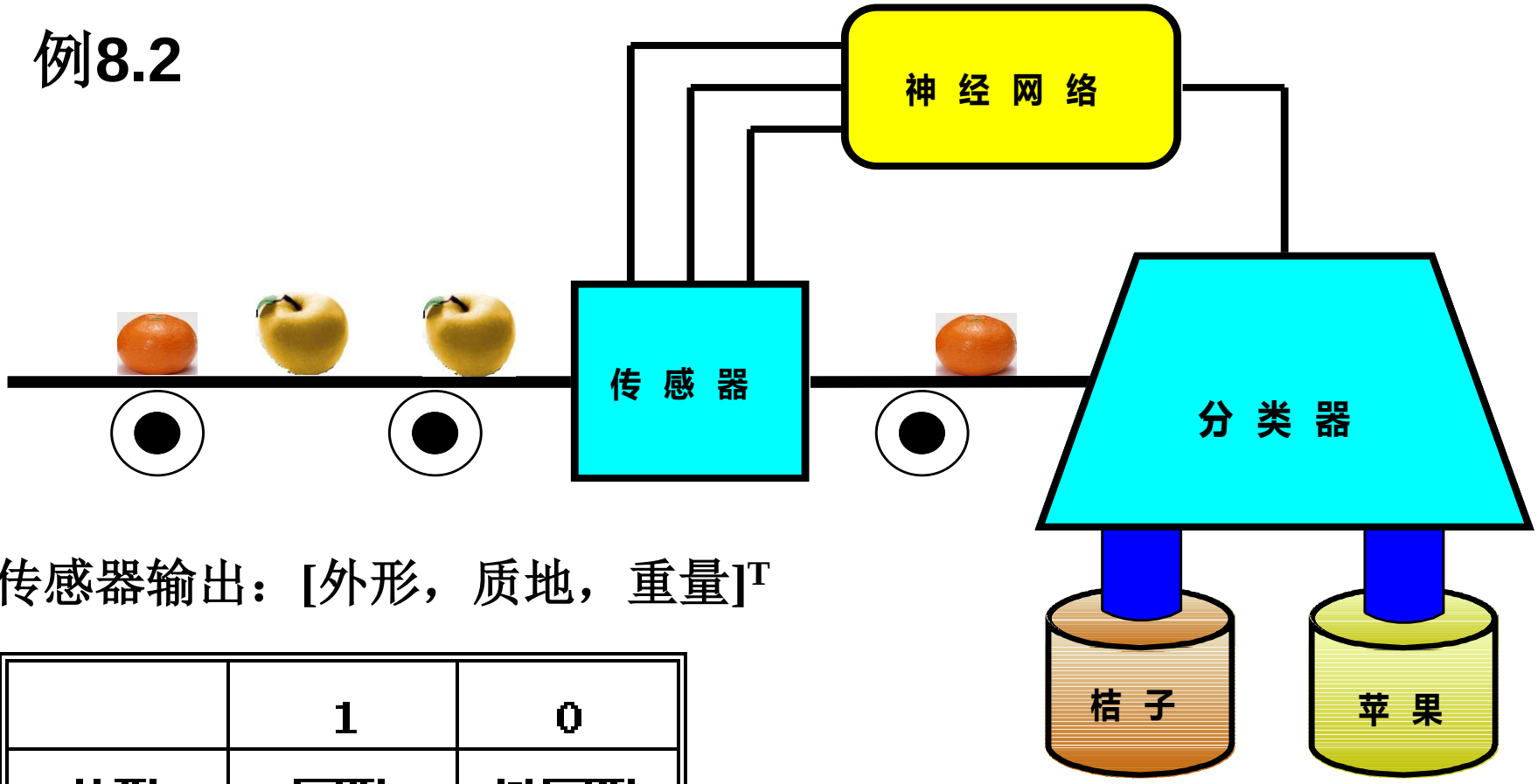
8.5.1 Hopfield神经网络在联想记忆中的应用

如何实现HNN的联想记忆功能?

- 网络能够通过联想来输出和输入模式最为相似的样本模式。

8.5.1 Hopfield神经网络在联想记忆中的应用

例8.2



- 传感器输出：[外形，质地，重量]^T

	1	0
外形	圆形	椭圆形
质地	光滑	粗糙
重量	< 1 磅	> 1 磅

标准桔子：$x^{(1)} = [1 \quad 0 \quad 1]^T$

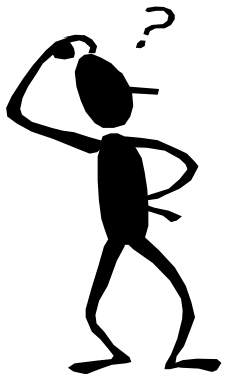
标准苹果：$x^{(2)} = [0 \quad 1 \quad 0]^T$

8.5.1 Hopfield神经网络在联想记忆中的应用

例8.2

- **样本：** $x^{(1)} = [1, 0, 1]^T$ (桔子)　　$x^{(2)} = [0, 1, 0]^T$（苹果）

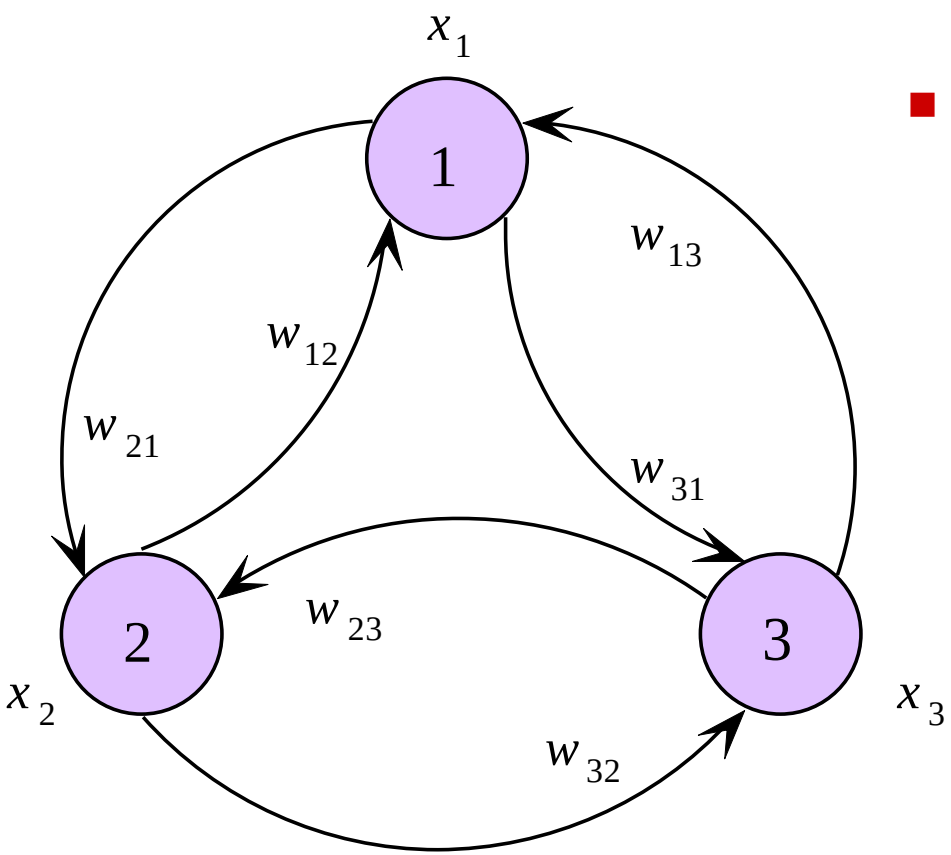
具体怎样实现联想记忆？



- **步骤：**
  - （1）设计DHNN结构
  - （2）设计连接权矩阵
  - （3）测试

8.5.1 Hopfield神经网络在联想记忆中的应用

- **（1）设计DHNN结构**



3神经元的DHNN结构图

- **样本：** $x^{(1)} = [1,\ 0,\ 1]^T$（桔子）

  $x^{(2)} = [0, 1,\ 0]^T$（苹果）

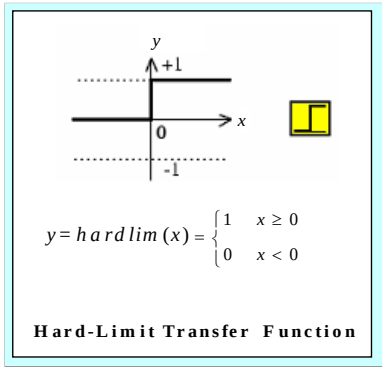
注： $\theta_i = 0 (i = 1,2,3)$

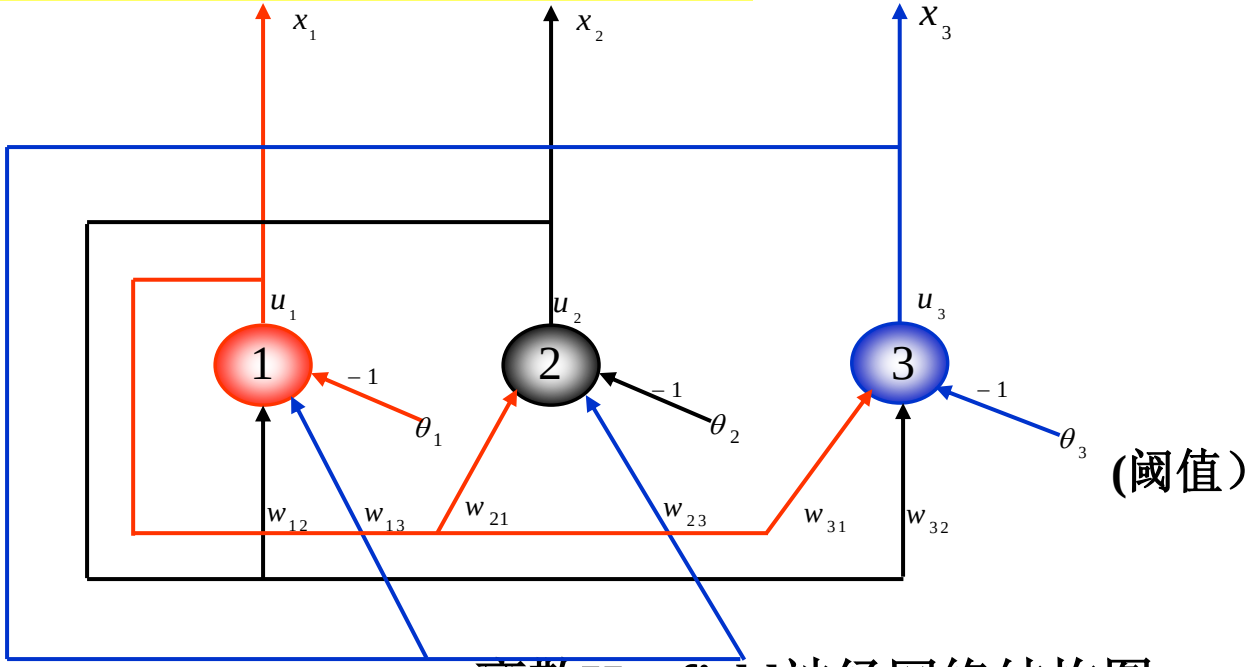
8.5.1 Hopfield神经网络在联想记忆中的应用

$$u_i(k+1) = \sum_{\substack{j=1 \\ j\neq i \\ j\to i}}^{N} w_{ij}x_j(k) - \theta_i, \quad w_{ij} = \begin{cases} 0 & i=j \end{cases}$$

$$x_i(k+1) = f(u_i(k+1)) = f\left(\sum_{\substack{j=1 \\ j\neq i \\ j\to i}}^{N} w_{ij}x_j(k) - \theta_i\right)$$

f是硬极限函数





(阈值)

离散Hopfield神经网络结构图

8.5.1 Hopfield神经网络在联想记忆中的应用

- （2）设计连接权矩阵

- 样本：$x^{(1)}=[1,\ 0,\ 1]^{\mathrm{T}}$（桔子）　$x^{(2)}=[0,\ 1,\ 0]^{\mathrm{T}}$（苹果）

- 连接权：$$w_{ij}=\begin{cases}\sum_{l=1}^{2}(2x_i^{(l)}-1)(2x_j^{(l)}-1) & i\neq j \quad l\text{是第}l\text{个样本}\\ 0 & i=j\end{cases}$$

$w_{ji}=w_{ij}(i=1,2,\ldots,N;\ j=1,2,\ldots,N)$ 串行，要求w是对称阵

$$w_{12}=(2x_1^{(1)}-1)(2x_2^{(1)}-1)+(2x_1^{(2)}-1)(2x_2^{(2)}-1)$$

$$=(2\times1-1)\times(2\times0-1)+(2\times0-1)\times(2\times1-1)$$

$$=-1-1=-2$$

$$w_{21}=w_{12}=-2$$

8.5.1 Hopfield神经网络在联想记忆中的应用

- （2）设计连接权矩阵

- 样本：$x^{(1)} = [1, 0, 1]^T$　　$x^{(2)} = [0, 1, 0]^T$

- 连接权：$$w_{ij} = \begin{cases} \sum_{l=1}^{2} (2x_i^{(l)} - 1)(2x_j^{(l)} - 1) & i \neq j \\ 0 & i = j \end{cases}$$

$$w_{ji} = w_{ij} (i = 1, 2, ..., N; j = 1, 2, ..., N)$$

$$w_{23} = (2x_2^{(1)} - 1)(2x_3^{(1)} - 1) + (2x_2^{(2)} - 1)(2x_3^{(2)} - 1)$$

$$= (2 \times 0 - 1) \times (2 \times 1 - 1) + (2 \times 1 - 1) \times (2 \times 0 - 1)$$

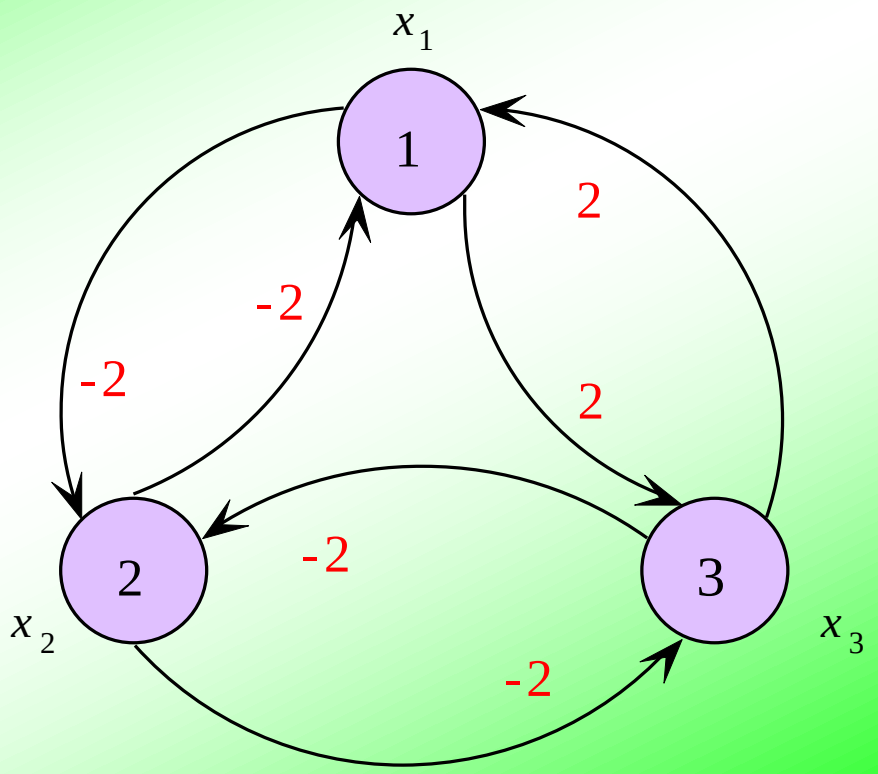
$$= -1 + (-1) = -2$$

$$w_{32} = w_{23} = -2$$

8.5.1 Hopfield神经网络在联想记忆中的应用

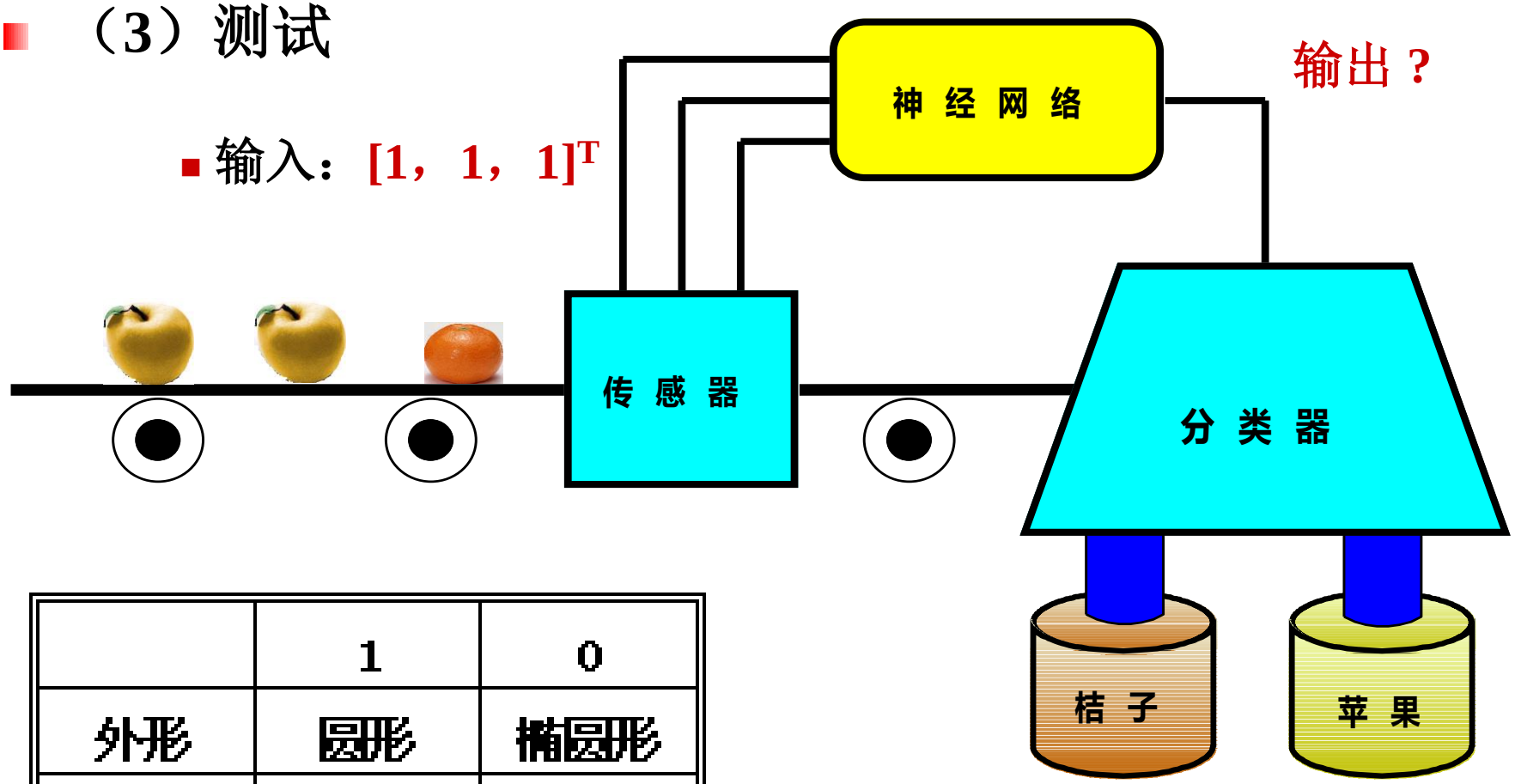
- （2）设计连接权矩阵

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



8.5.1 Hopfield神经网络在联想记忆中的应用

- （3）测试
  - 输入：$[1，1，1]^T$



	1	0
外形	圆形	椭圆形
质地	光滑	粗糙
重量	＜1 磅	＞1 磅

标准桔子：$x^{(1)} = [1, 0, 1]^T$

标准苹果：$x^{(2)} = [0, 1, 0]^T$

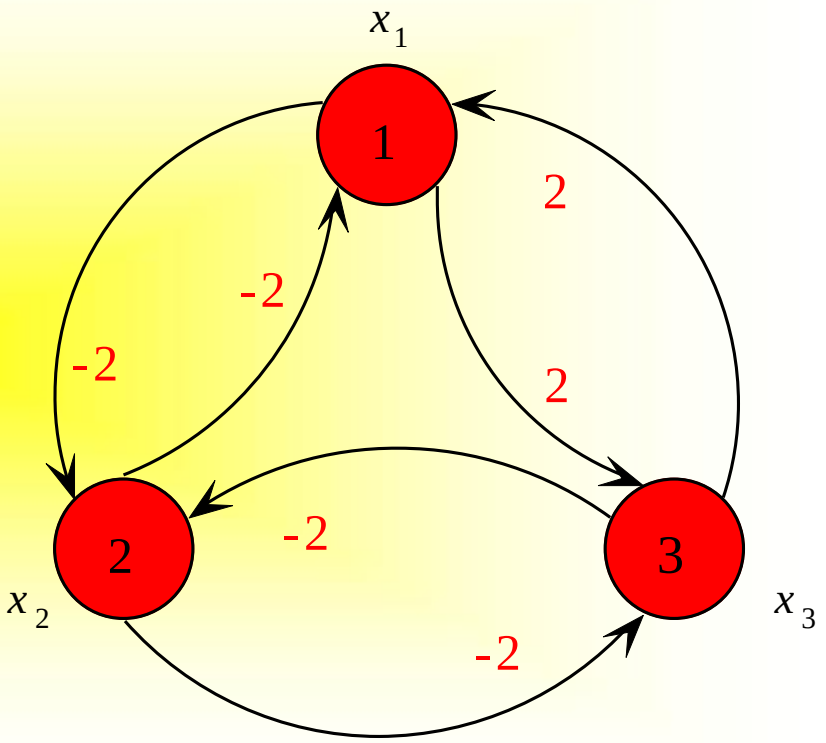
8.5.1 Hopfield神经网络在联想记忆中的应用

（3）测试

- 样本：$x^{(1)}=[1,\ 0,\ 1]^{T}$（桔子）　　$x^{(2)}=[0,1,\ 0]^{T}$（苹果）
- 测试用例：$[1,\ 1,\ 1]^{T}$
- 初始状态：

$$\begin{cases} x_1(0)=1 \\ x_2(0)=1 \\ x_3(0)=1 \end{cases}$$

- 调整次序：$2 \rightarrow 1 \rightarrow 3$

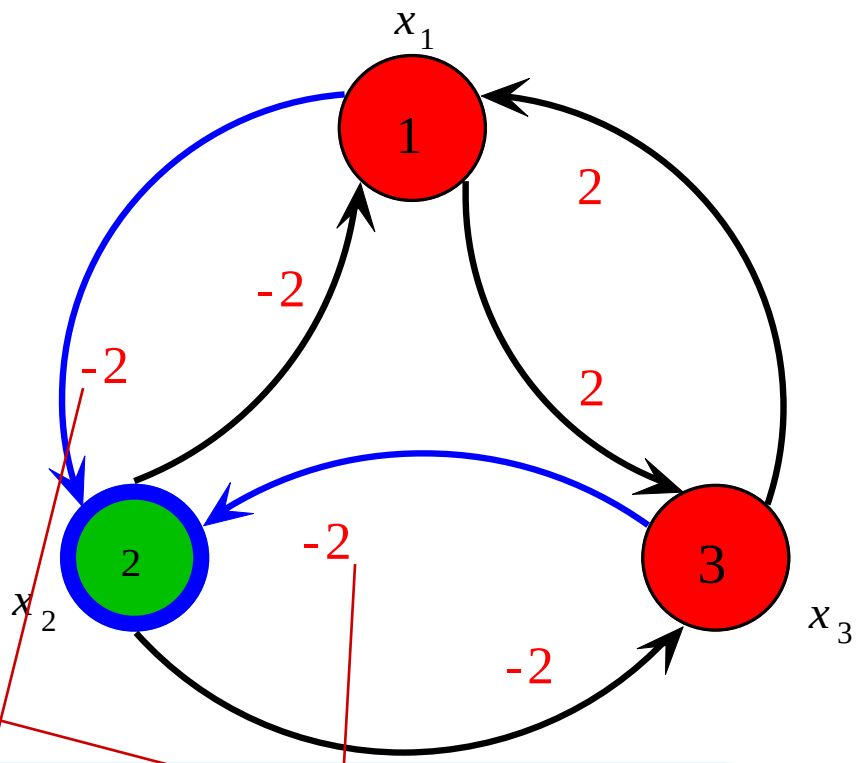


8.5.1 Hopfield神经网络在联想记忆中的应用

- 调整次序： 2→1→3

t = 0

$$\begin{cases} x_1(0)=1 \\ x_2(0)=1 \\ x_3(0)=1 \end{cases}$$



$$x_2(1)=f\left(\sum_{j=1}^{3} w_{2j}x_j(0)\right)=f((-2)\times 1+0\times 1+(-2)\times 1)=f(-4)=0$$

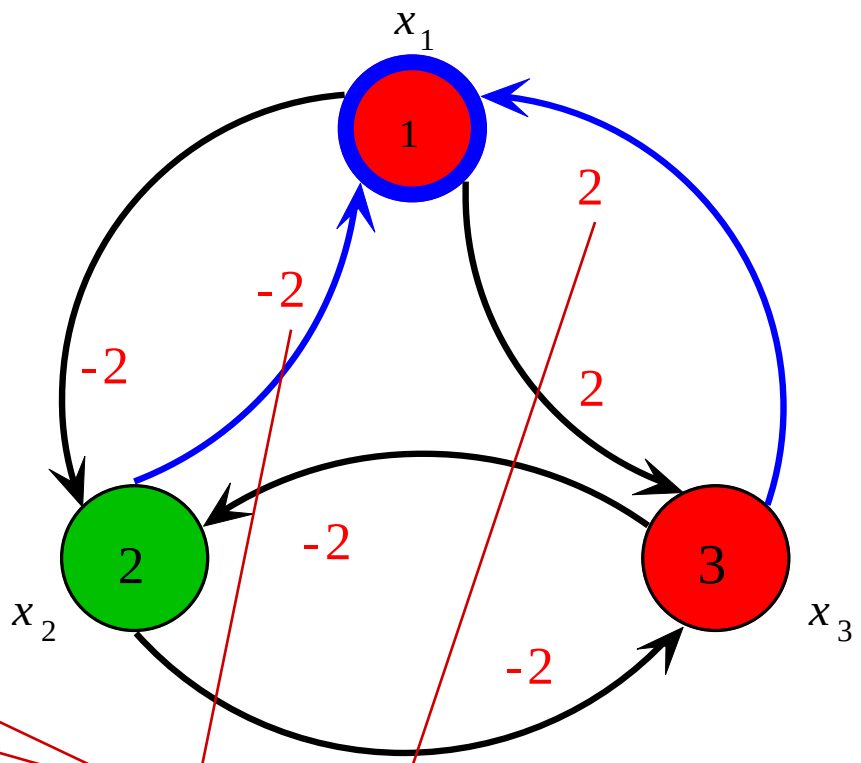
$$x_1(1)=x_1(0)=1,\quad x_3(1)=x_3(0)=1$$

8.5.1 Hopfield神经网络在联想记忆中的应用

- 调整次序： 2→1→3

$t = 1$

$$\begin{cases} x_1(0) = 1 \\ x_2(0) = 0 \\ x_3(0) = 1 \end{cases}$$



$$x_1(2) = f\left(\sum_{j=1}^{3} w_{1j}x_j(1)\right) = f(0\times 1 + (-2)\times 0 + 2\times 1) = f(2) = 1$$

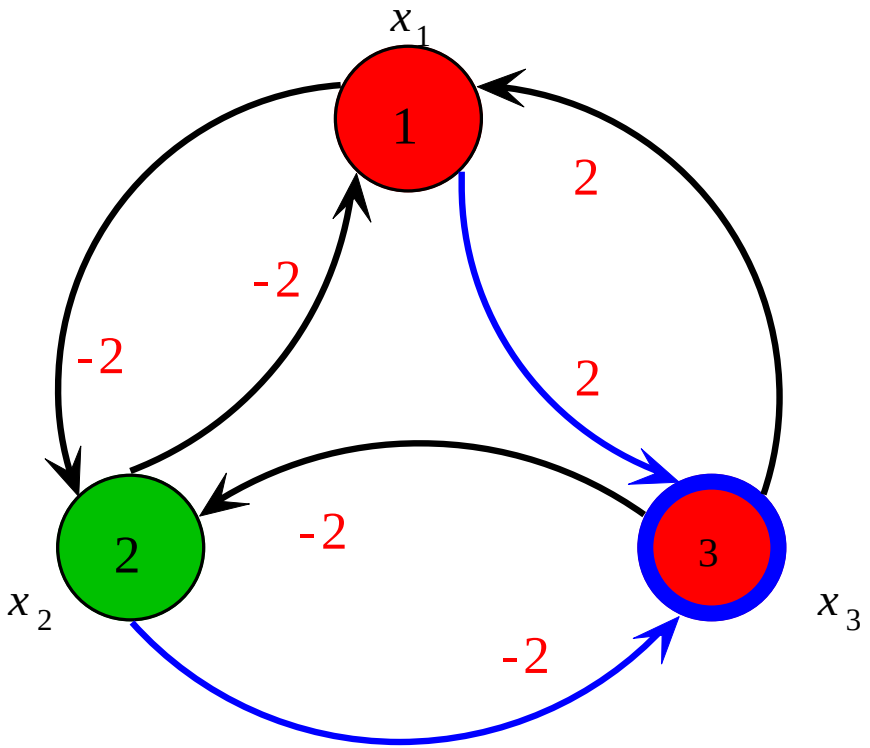
$$x_2(2) = x_2(1) = 0, \qquad x_3(2) = x_3(1) = 1$$

8.5.1 Hopfield神经网络在联想记忆中的应用

- 调整次序： 2→1→3

t = 2

$$\begin{cases} x_1(0) = 1 \\ x_2(0) = 0 \\ x_3(0) = 1 \end{cases}$$



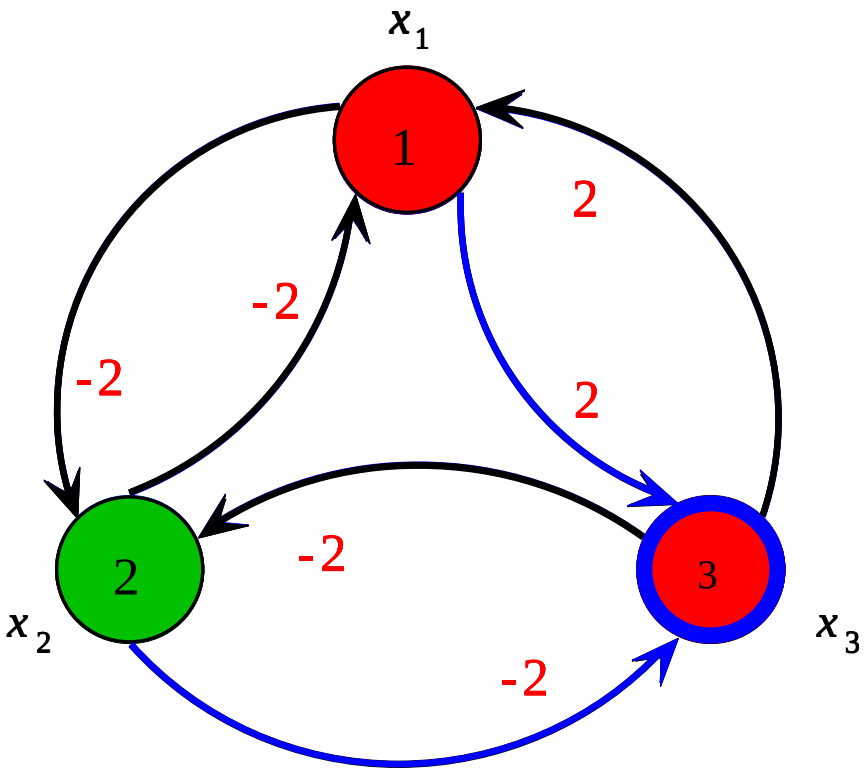
$$x_3(3) = f\left(\sum_{j=1}^{3} w_{3j}x_j(1)\right) = f(2\times 1 + (-2)\times 0 + 0\times 1) = f(2) = 1$$

$$x_1(3) = x_1(2) = 1, \qquad x_2(3) = x_2(2) = 0$$

8.5.1 Hopfield神经网络在联想记忆中的应用

- 调整次序： **2 → 1 → 3**
- 样本： $x^{(1)} = [1,\ 0,\ 1]^T$ (桔子)

$x^{(2)} = [0,\ 1,\ 0]^T$ (苹果)



t = 0

$$\begin{cases} x_1(0) = 1 \\ x_2(0) = 1 \\ x_3(0) = 1 \end{cases}$$

t = 1

$$\begin{cases} x_1(1) = 1 \\ x_2(1) = 0 \\ x_3(1) = 1 \end{cases}$$

t = 2

$$\begin{cases} x_1(2) = 1 \\ x_2(2) = 0 \\ x_3(2) = 1 \end{cases}$$

t = 3

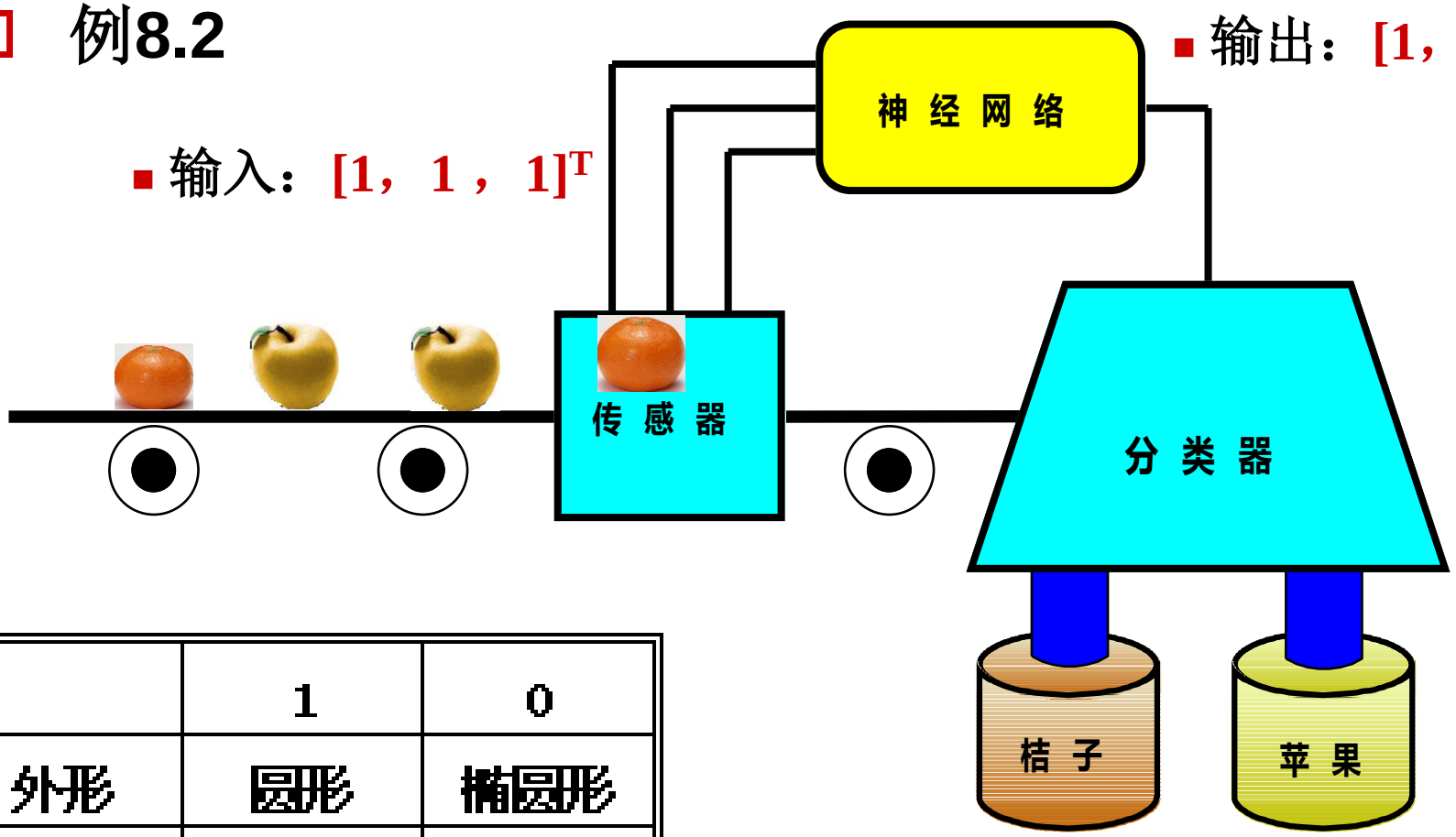
$$\begin{cases} x_1(3) = 1 \\ x_2(3) = 0 \\ x_3(3) = 1 \end{cases}$$

8.5.1 Hopfield神经网络在联想记忆中的应用

- 例8.2

- 输入：$[1，1，1]^T$

- 输出：$[1，0，1]^T$



	1	0
外形	圆形	椭圆形
质地	光滑	粗糙
重量	< 1 磅	> 1 磅

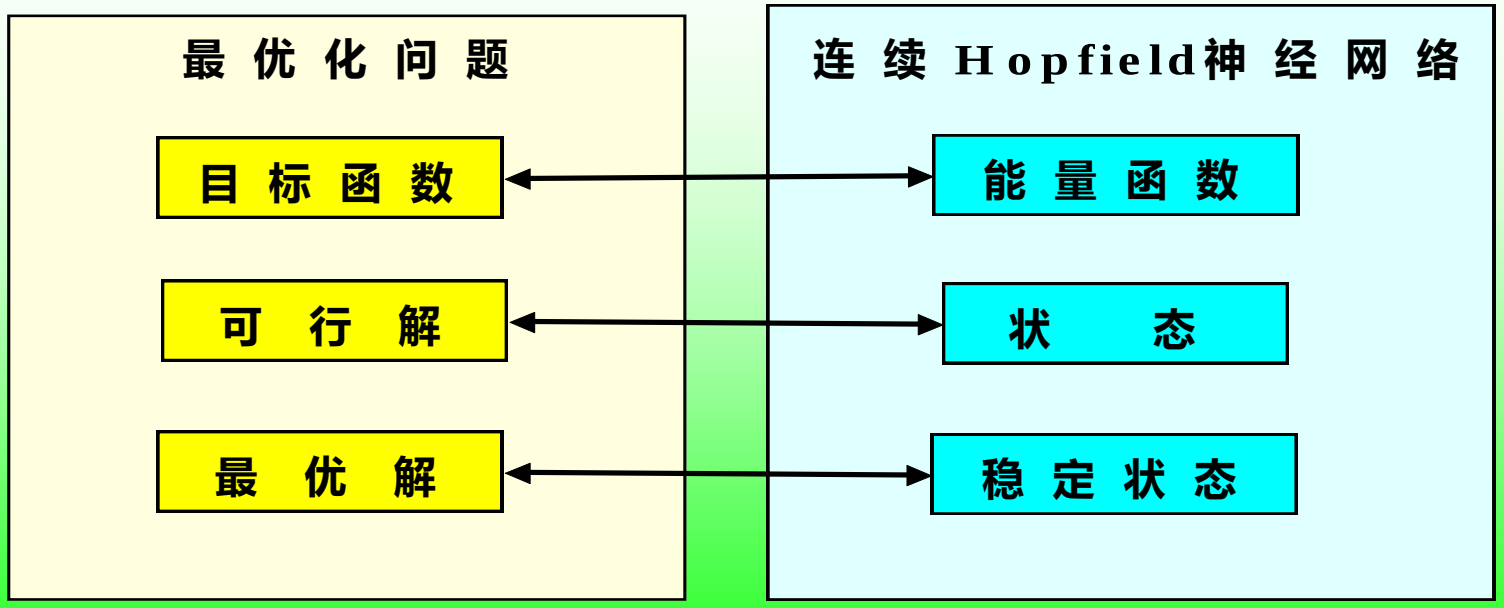
标准桔子：$x^{(1)} = [1, 0, 1]^T$

标准苹果：$x^{(2)} = [0, 1, 0]^T$

8.5.2 Hopfield神经网络优化方法

- **1985年，霍普菲尔德和塔克（D. W. Tank）应用连续Hopfield神经网络求解旅行商问题（traveling salesman problem，TSP）获得成功。**

- **连续Hopfiled神经网络求解约束优化问题的基本思路：**



8.5.2 Hopfield神经网络优化方法

- 用神经网络方法求解优化问题的一般步骤：

（1）将优化问题的每一个可行解用换位矩阵表示。

（2）将换位矩阵与由*N*个神经元构成的神经网络相对应：每一个可行解的换位矩阵的各元素与相应的神经元稳态输出相对应。

（3）构造能量函数，使其最小值对应于优化问题的最优解，并满足约束条件。

（4）用罚函数法构造目标函数，与Hopfield神经网络的计算能量函数表达式相等，确定各连接权和偏置参数。

（5）给定网络初始状态和网络参数等，使网络按动态方程运行，直到稳定状态，并将它解释为优化问题的解。

8.5.2 Hopfield神经网络优化方法

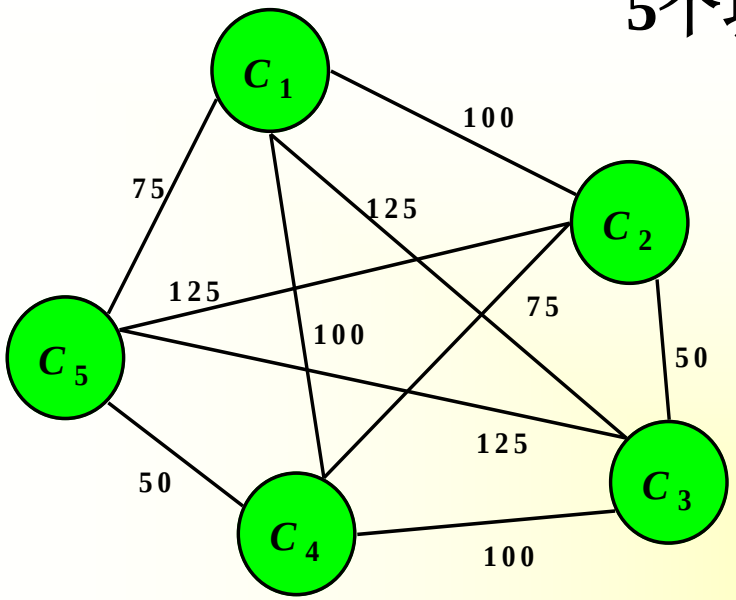
- 应用举例： **Hopfield神经网络优化方法求解TSP。**

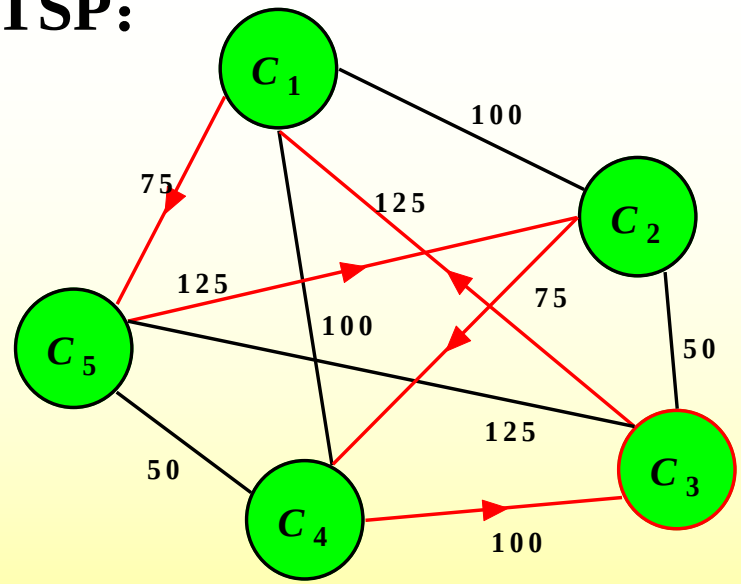
- **1985年，霍普菲尔德和塔克（D. W. Tank）应用连续Hopfield 神经网络求解旅行商问题获得成功。**

- **旅行商问题（traveling salesman problem，TSP）：有 *n* 个城市，城市间的距离或旅行成本已知，求合理的路线使每个城市都访问一次，且总路径（或者总成本）为最短。**

8.5.2 Hopfield神经网络优化方法

5个城市的TSP：





	1	2	3	4	5
C_1	0	1	0	0	0
C_2	0	0	0	1	0
C_3	1	0	0	0	0
C_4	0	0	0	0	1
C_5	0	0	1	0	0

神经元数目：25

8.5.2 Hopfield神经网络优化方法

- **TSP的描述：**

$$\min \quad l = \frac{1}{2}\sum_{x}\sum_{y\neq x}\sum_{i} d_{xy} v_{xi}(v_{y,i+1} + v_{y,i-1})$$

$$st. \quad \sum_{x} v_{xi} = 1 \qquad \forall i \qquad \text{（每次只能访问一个城市）}$$

$$\sum_{i} v_{xi} = 1 \qquad \forall x \qquad \text{（每个城市只能访问一次）}$$

- **用罚函数法，写出优化问题的目标函数：**

$$J = \frac{A}{2}\sum_{x}\sum_{i}\sum_{j\neq i} v_{xi}v_{xj} + \frac{B}{2}\sum_{i}\sum_{x}\sum_{y\neq x} v_{xj}v_{yi} + \frac{C}{2}\left(\sum_{x}\sum_{i} v_{xi} - N\right)^2 +$$

$$\frac{D}{2}\sum_{x}\sum_{y\neq x}\sum_{i} d_{xy} v_{xi}(v_{y,i+1} + v_{y,i-1})$$

8.5.2 Hopfield神经网络优化方法

- **Hopfield神经网络能量函数：**

$$E = -\frac{1}{2}\sum_{x=1}^{N}\sum_{i=1}^{N}\sum_{y=1}^{N}\sum_{j=1}^{N} w_{xi,yj} v_{xi} v_{yj} - \sum_{x=1}^{N}\sum_{i=1}^{N} v_{xi} I_{xi} + \sum_{x=1}^{N}\sum_{i=1}^{N} \frac{1}{R'_{xi}} \int_0^{v_{xi}} f^{-1}(v)\mathrm{d}v$$

$$= E_1 + \sum_{x=1}^{N}\sum_{i=1}^{N} \frac{1}{R'_{xi}} \int_0^{v_{xi}} f^{-1}(v)\mathrm{d}v$$

- **令E_1与目标函数J相等，确定神经网络的连接权值和偏置电流：**

$$W_{xi,yj} = A\delta_{xy}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{xy}) - C - D\delta_{xy}(\delta_{j,i+1} + \delta_{j,i-1})$$

$$I_{xi} = -CN$$

8.5.2 Hopfield神经网络优化方法

- 神经网络的动态方程：

$$C_{xi}\frac{du_{xj}}{dt}=-\frac{\partial E}{\partial v_{xi}}=-\frac{\partial\left(E_1+\sum_{x=1}^{N}\sum_{i=1}^{N}\frac{1}{R'_{xi}}\int_0^{v_{xi}}f^{-1}(v)\mathrm{d}v\right)}{\partial v_{xi}}$$

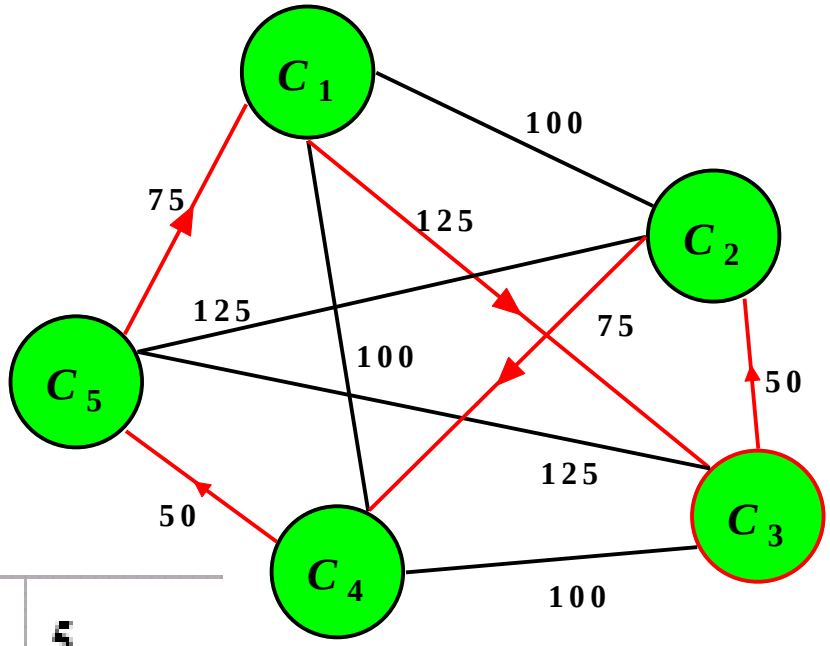
$$=-\frac{\partial\left(J+\sum_{x=1}^{N}\sum_{i=1}^{N}\frac{1}{R'_{xi}}\int_0^{v_{xi}}f^{-1}(v)\mathrm{d}v\right)}{\partial v_{xi}}$$

$$=-\frac{u_{xi}}{R'_{xi}}-A\sum_{j\neq i}v_{xj}-B\sum_{y\neq x}v_{yi}-C\left(\sum_{x}\sum_{i}v_{xi}-N\right)-D\sum_{y}d_{xy}\left(v_{y,i+1}+v_{y,i-1}\right)$$

$$v_{xi}=f(u_{xi})=\frac{1}{2}\left[1+\tanh\left(\frac{u_{xi}}{u_0}\right)\right]$$

8.5.2 Hopfield神经网络优化方法

- 选择合适的A、B、C、D和网络的初始状态，按网络动态方程演化直到收敛。



	1	2	3	4	5
C_1	0	0	0	0	1
C_2	0	1	0	0	0
C_3	1	0	0	0	0
C_4	0	0	1	0	0
C_5	0	0	0	1	0

8.5.2 Hopfield神经网络优化方法

- 神经网络优化计算目前存在的问题：

（1）解的不稳定性。

（2）参数难以确定。

（3）能量函数存在大量局部极小值，难以保证最优解。

THE END