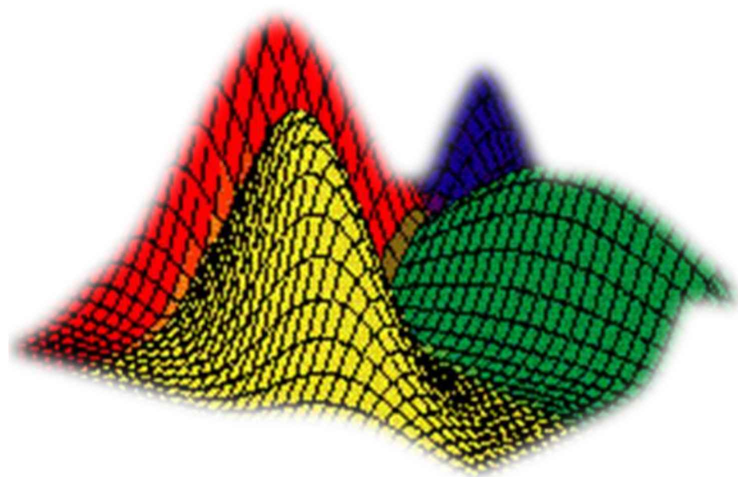


CSE 473
Pattern Recognition

Context Dependent Classification

Context Dependent Classification

- Recall context free classification
  - No relation exist among classes
  - No relation exists among objects (feature vectors)
  - A new object is classified to any class independent of the previous objects' classes

Context Dependent Classification

- In Context dependent classification, the class of a feature vector depends on
  - Its own value
  - Value of other feature vectors
  - Classes assigned to other vectors

Context Dependent Classification

- Application
  - Communication
  - Image Processing
  - Signal Processing

Solution for Context Dependent Classification

- Recall Bayesian formulation for context free classification
  - Assign x to ω_i if $P(\omega_i|\underline{x}) > P(\omega_j|\underline{x}),\ \forall j \neq i$
- In context dependent classification, we cannot apply it directly because of interdependency of features and classes

Solution for Context Dependent Classification

- This interrelation demands the classification to be performed simultaneously for all available feature vectors

- we assume that the training vectors $\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_N$ occur in sequence, one after the other and we will refer to them as **observations**

Context Dependent Bayesian Classifier

- Let $X:\{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_N\}$ be sequence of observations

- Let $\omega_i,\ i = 1,2,\ldots,M$ be the available *M* classes

- Let Ω_i be a possible sequence of assigned classes, that is

$$\Omega_i : \omega_{i1}\ \omega_{i2}\ \ldots \omega_{iN}$$

where, $i_k \in \{1, 2, \ldots, M\}$ for $k = 1, 2, \ldots, N$

- There are M^N of Ω_i

Context Dependent Bayesian Classifier

- Now, given $X:\{\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_N\}$ and $\Omega_i : \omega_{i1}\, \omega_{i2} \ldots \omega_{iN}$

  Classify X to using the Bayesian rule

$$\boxed{X \to \Omega_i : \quad P(\Omega_i \mid X) > P(\Omega_j \mid X) \;\; \forall i \neq j, \quad i, j = 1, 2, \ldots, M^N}$$

- This is equivalent to classifying

$$\boldsymbol{x}_1 \text{ to class } \omega_{i_1}, \boldsymbol{x}_2 \text{ to } \omega_{i_2}, \text{and so on}$$

Context Dependent Bayesian Classifier

- The rule

$$\boxed{P(\Omega_i | X) > P(\Omega_j | X) \ \ \forall i \neq j}$$

can be simplified as

$$P(\Omega_i)p(X|\Omega_i) > P(\Omega_j)p(X|\Omega_j), \quad \forall i \neq j$$

Further Simplification: Markov Chain Model

$$P(\Omega_i)p(X|\Omega_i) > P(\Omega_j)p(X|\Omega_j), \quad \forall i \neq j$$

- Markov Chain Models (for class dependence)

$$P(\omega_{i_k} | \omega_{i_{k-1}}, \omega_{i_{k-2}}, \ldots, \omega_{i_1}) = P(\omega_{i_k} | \omega_{i_{k-1}})$$

which means class dependence is limited to only within two successive classes

Further Simplification: Markov Chain Model

- Markov Chain Models (for class dependence)

$$P(\omega_{i_k} | \omega_{i_{k-1}}, \omega_{i_{k-2}}, \ldots, \omega_{i_1}) = P(\omega_{i_k} | \omega_{i_{k-1}})$$

in other words,

if observations $\boldsymbol{x}_{k-1}, \boldsymbol{x}_{k-2}, \ldots, \boldsymbol{x}_1$ belong to classes $\omega_{i_{k-1}}, \omega_{i_{k-2}}, \ldots, \omega_{i_1}$

then observation $\boldsymbol{x}_k$, at stage k, belonging to class ω_{i_k}

depends on the *class from which observation* $\boldsymbol{x}_{k-1}$*, at stage k-1 has occurred*

Further Simplification: Markov Chain Model

- Markov Chain Models (for class dependence)

$$P(\omega_{i_k} | \omega_{i_{k-1}}, \omega_{i_{k-2}}, \ldots, \omega_{i_1}) = P(\omega_{i_k} | \omega_{i_{k-1}})$$

- Therefore, we can write

$$P(\Omega_i) \equiv P(\omega_{i_1}, \omega_{i_2}, \ldots, \omega_{i_N})$$

$$= P(\omega_{i_N} | \omega_{i_{N-1}}, \ldots, \omega_{i_1}) P(\omega_{i_{N-1}} | \omega_{i_{N-2}}, \ldots, \omega_{i_1}) \ldots P(\omega_{i_1})$$

Further Simplification: Markov Chain Model

- Markov Chain Models (for class dependence)

$$P(\omega_{i_k} | \omega_{i_{k-1}}, \omega_{i_{k-2}}, \ldots, \omega_{i_1}) = P(\omega_{i_k} | \omega_{i_{k-1}})$$

- Therefore, we can write

$$P(\Omega_i) \equiv P(\omega_{i_1}, \omega_{i_2}, \ldots, \omega_{i_N})$$
$$= P(\omega_{i_N} | \omega_{i_{N-1}}, \ldots, \omega_{i_1}) P(\omega_{i_{N-1}} | \omega_{i_{N-2}}, \ldots, \omega_{i_1}) \ldots P(\omega_{i_1})$$

- We find,

$$P(\Omega_i) = P(\omega_{i_1}) \prod_{k=2}^{N} P(\omega_{i_k} | \omega_{i_{k-1}})$$

Further Simplification: Markov Chain Model

- Further assumption
  - $\underline{x}_i$ statistically mutually independent

$$
\boxed{
\begin{aligned}
p(X|\Omega_i) &= p(\underline{x}_1, \underline{x}_2, \underline{x}_3 \cdots, \underline{x}_N|\Omega_i) \\
&= p(\underline{x}_1|\Omega_i)\, p(\underline{x}_2|\Omega_i)\, p(\underline{x}_3|\Omega_i) \cdots p(\underline{x}_N|\Omega_i) \\
&= \prod_{k=1}^{N} p(\underline{x}_k|\Omega_i)
\end{aligned}
}
$$

Further Simplification: Markov Chain Model

- Further assumption
  - $\underline{x}_i$ statistically mutually independent

$$
\boxed{
\begin{aligned}
p(X|\Omega_i) &= p(\underline{x}_1, \underline{x}_2, \underline{x}_3 \cdots, \underline{x}_N|\Omega_i) \\
&= p(\underline{x}_1|\Omega_i)\, p(\underline{x}_2|\Omega_i)\, p(\underline{x}_3|\Omega_i) \cdots p(\underline{x}_N|\Omega_i) \\
&= \prod_{k=1}^{N} p(\underline{x}_k|\Omega_i)
\end{aligned}
}
$$

  - If the pdf in one class is independent of the others, then

$$p(\vec{x}_k \mid \Omega_i) = p(\vec{x}_k \mid \omega_{i1}\, \omega_{i2} \ldots \omega_{iN}) = p(\vec{x}_k \mid \omega_{ik})$$

Further Simplification: Markov Chain Model

- Further assumption
  - $\underline{x}_i$ statistically mutually independent

$$\boxed{\begin{aligned} p(X|\Omega_i) &= p(\underline{x}_1, \underline{x}_2, \underline{x}_3 \cdots, \underline{x}_N|\Omega_i) \\ &= p(\underline{x}_1|\Omega_i)\,p(\underline{x}_2|\Omega_i)\,p(\underline{x}_3|\Omega_i)\cdots p(\underline{x}_N|\Omega_i) \\ &= \prod_{k=1}^{N} p(\underline{x}_k|\Omega_i) \end{aligned}}$$

  - The pdf in one class independent of the others, then

$$p(\vec{x}_k \mid \Omega_i) = p(\vec{x}_k \mid \omega_{i1}\ \omega_{i2} \ldots \omega_{iN}) = p(\vec{x}_k \mid \omega_{ik})$$

Finally,

$$\boxed{p(X|\Omega_i) = \prod_{k=1}^{N} p(\underline{x}_k|\omega_{i_k})}$$

Further Simplification: Markov Chain Model

- From the above, the Bayes rule is readily seen to be equivalent to:

$$P(\Omega_i | X) \quad (><) \quad P(\Omega_j | X)$$

$$P(\Omega_i) p(X | \Omega_i) \quad (><) \quad P(\Omega_j) p(X | \Omega_j)$$

Further Simplification: Markov Chain Model

- From the above, the Bayes rule is readily seen to be equivalent to:

$$P(\Omega_i | X) \quad (><) \quad P(\Omega_j | X)$$

$$P(\Omega_i) p(X | \Omega_i) \quad (><) \quad P(\Omega_j) p(X | \Omega_j)$$

that is, it turns to:

$$\boxed{\begin{aligned} & p(X | \Omega_i) P(\Omega_i) = P(\omega_{i_1}) p(\underline{x}_1 | \omega_{i_1}). \\ & \prod_{k=2}^{N} P(\omega_{i_k} | \omega_{i_{k-1}}) p(\underline{x}_k | \omega_{i_k}) \end{aligned}}$$

Further Simplification: Markov Chain Model

- From the above, the Bayes rule is readily seen to be equivalent to:

$$P(\Omega_i|X) \quad (><) \quad P(\Omega_j|X)$$

$$P(\Omega_i)p(X|\Omega_i) \quad (><) \quad P(\Omega_j)p(X|\Omega_j)$$

that is, it rests on

$$\boxed{\begin{aligned} &p(X|\Omega_i)P(\Omega_i) = P(\omega_{i_1})p(\underline{x}_1|\omega_{i_1}). \\ &\prod_{k=2}^{N} P(\omega_{i_k}|\omega_{i_{k-1}})p(\underline{x}_k|\omega_{i_k}) \end{aligned}}$$

- To find the above maximum in brute-force task we need $O(NM^N)$ operations!!

Further Simplification: Markov Chain Model

given a observation sequence

$$X = \left[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_N\right]$$

Let two class sequences

$$\Omega_i = \left[\omega_3, \omega_1, \omega_M, \cdots, \omega_1, \omega_9\right]$$

and $\Omega_j = \left[\omega_3, \omega_1, \omega_M, \cdots, \omega_1, \omega_8\right]$

Further Simplification: Markov Chain Model

given a observation sequence

$$X = [\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \mathbf{x}_N]$$

Let two class sequences

$$\Omega_i = [\omega_3, \omega_1, \omega_M, \cdots, \omega_1, \omega_9]$$

and $\Omega_j = [\omega_3, \omega_1, \omega_M, \cdots, \omega_1, \omega_8]$

$$\uparrow \quad \uparrow \quad \uparrow \qquad\qquad \uparrow$$

$$[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \qquad \mathbf{x}_N]$$

Further Simplification: Markov Chain Model

Let two class sequences

$$\Omega_i = [\omega_3, \omega_1, \omega_M, \cdots, \omega_1, \omega_9]$$

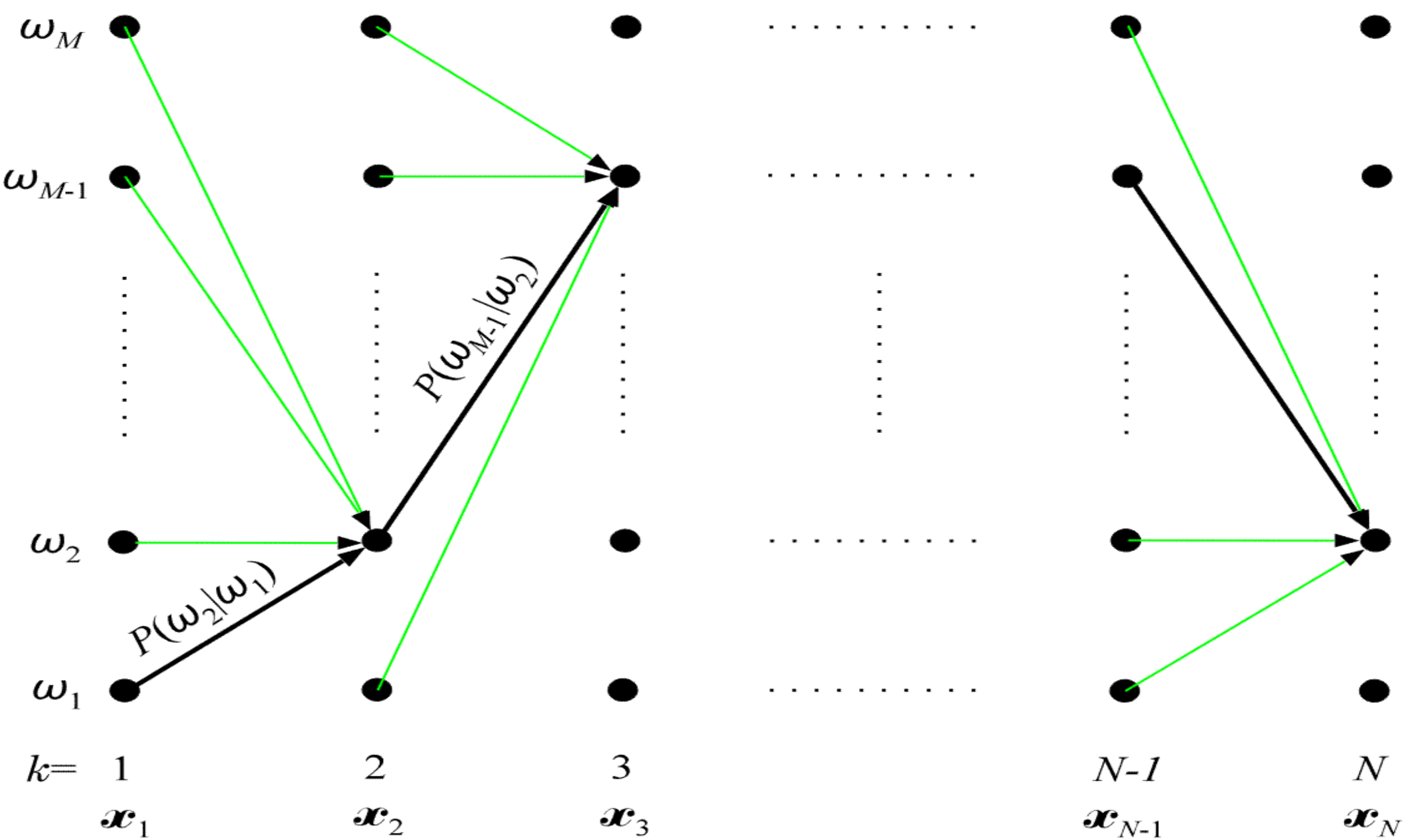
and $\Omega_j = [\omega_3, \omega_1, \omega_M, \cdots, \omega_1, \omega_8]$

$$[\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \cdots, \qquad \mathbf{x}_N]$$

$p(X|\Omega_i)P(\Omega_i)$ and $p(X|\Omega_j)P(\Omega_j)$ differ

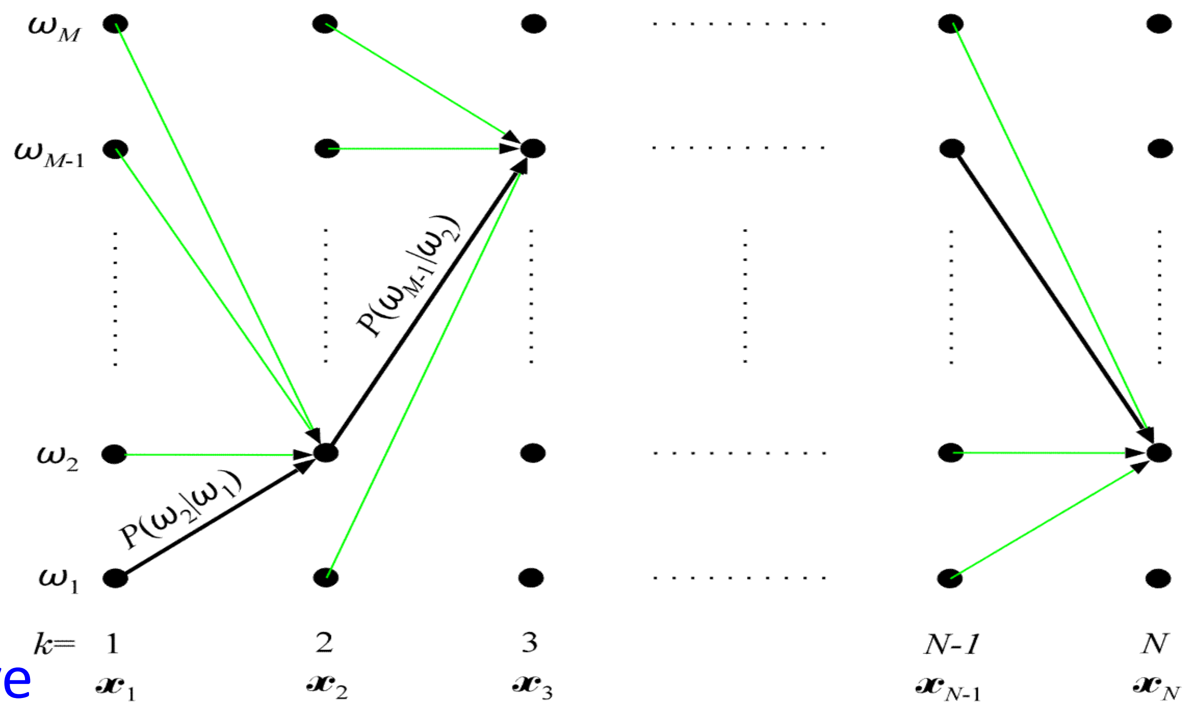
only in the last term

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$



Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

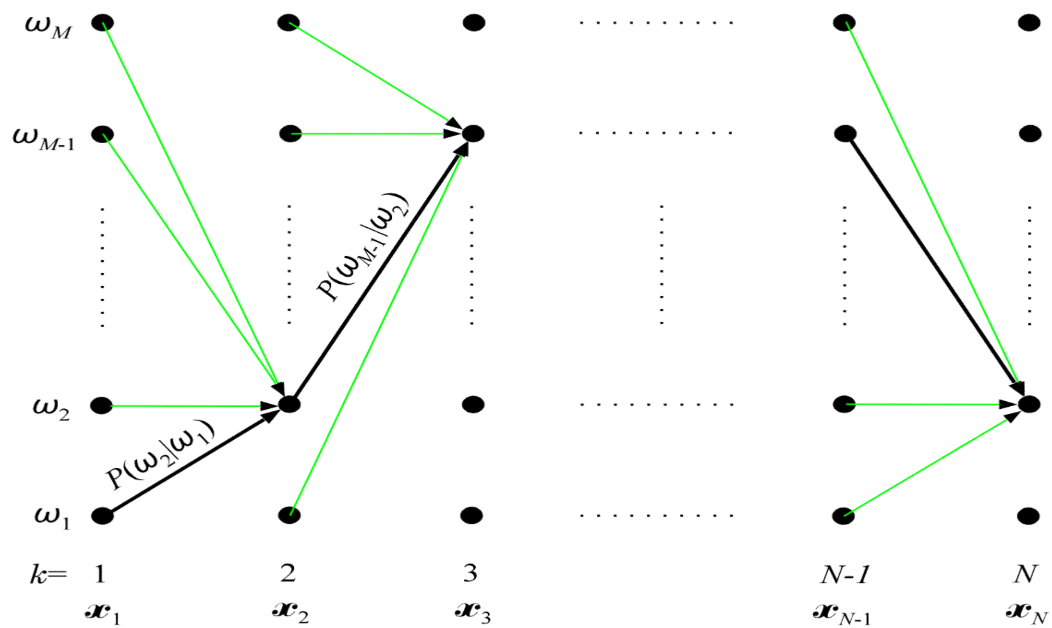
- *N* dot columns
- *M* possible classes
- Successive columns correspond to successive observations ***x**$_k$*, ***k*= 1, 2, …, *N***



- *Each of the class sequences Ω_i corresponds to a specific path of successive transitions*

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

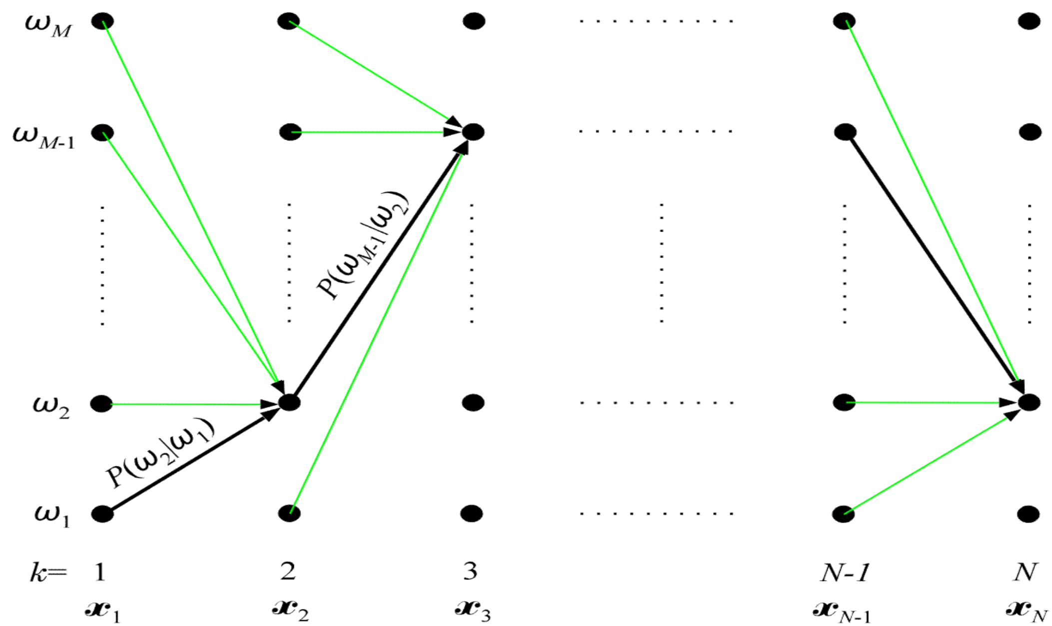
- Each transition from class *ω_i to ω_j has a fixed* probability *$P(\omega_j \mid \omega_i)$*
- *Also, assume $p(x|\ \omega_i)$ is known*



Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

- Now the problem can be stated as,

  Given a sequence of observations $x_1, x_2, \ldots, x_N$, *find the path of successive (class) transitions that* maximizes

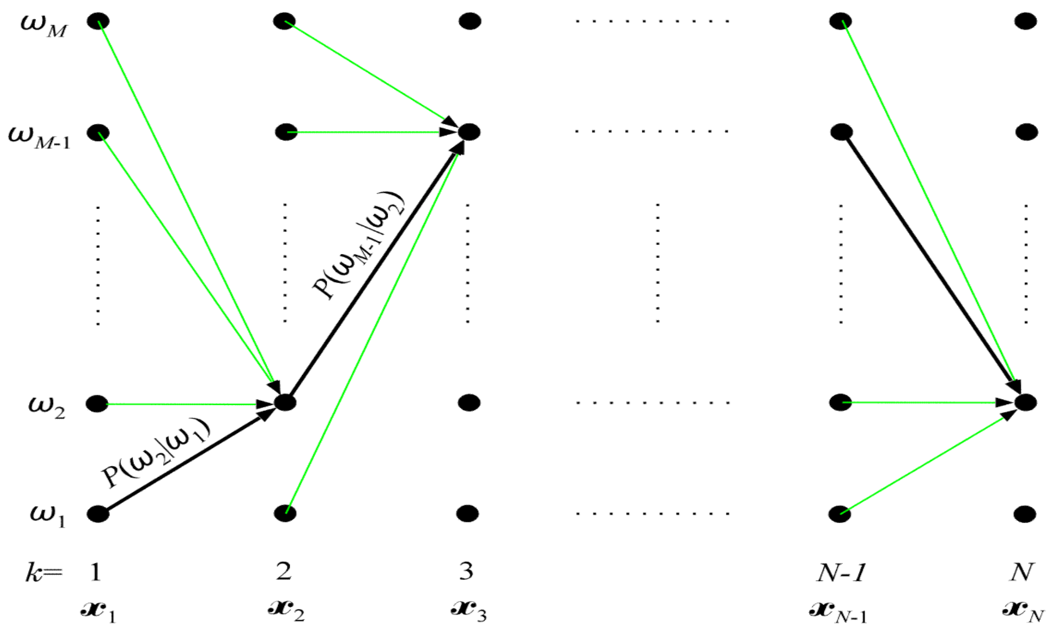


$$p(X|\Omega_i)P(\Omega_i) = P(\omega_{i_1})p(\underline{x}_1|\omega_{i_1}) \cdot \prod_{k=2}^{N} P(\omega_{i_k}|\omega_{i_{k-1}})p(\underline{x}_k|\omega_{i_k})$$

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

- Now the problem can be stated as,

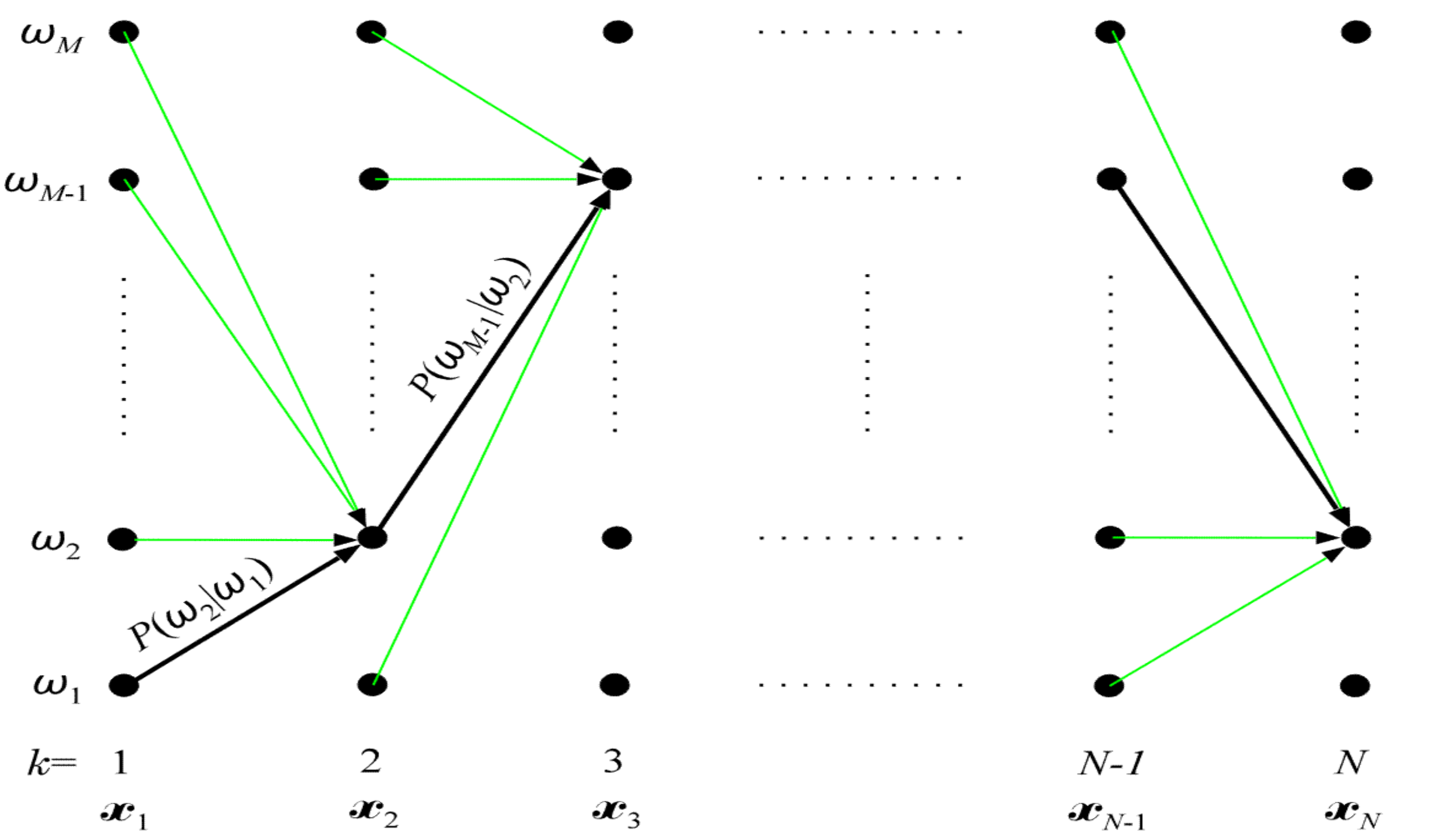
  Given a sequence of observations $x_1, x_2, \ldots, x_N$, *find the path of successive (class) transitions that* maximizes



$$p(X|\Omega_i)P(\Omega_i) = P(\omega_{i_1})p(\underline{x}_1|\omega_{i_1}) \cdot \prod_{k=2}^{N} P(\omega_{i_k}|\omega_{i_{k-1}})p(\underline{x}_k|\omega_{i_k})$$

- *That is, find an optimal path (e. g., the black line)*

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$



- *The classes along this optimal path are the classes of the respective observations.*

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

- Looking at

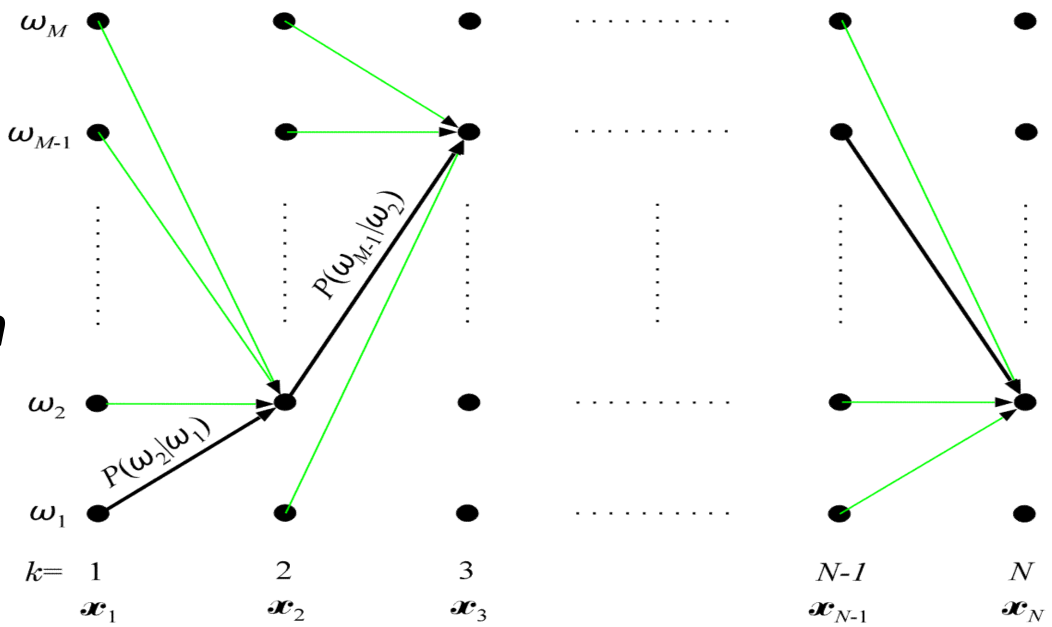
$$p(X|\Omega_i)P(\Omega_i) = P(\omega_{i_1})p(\underline{x}_1|\omega_{i_1}) \cdot \prod_{k=2}^{N} P(\omega_{i_k}|\omega_{i_{k-1}})p(\underline{x}_k|\omega_{i_k})$$

The cost of a transition is

$$\hat{d}(\omega_{i_k},\omega_{i_{k-1}}) = P(\omega_{i_k}|\omega_{i_{k-1}}).p(\underline{x}_k|\omega_{i_k})$$

And the cost at initial transition is

$$\hat{d}(\omega_{i_1},\omega_{i_0}) \equiv P(\omega_{i_1})p(\underline{x}_i|\omega_{i_1})$$



Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

- Using the notations

$$\hat{d}(\omega_{i_k}, \omega_{i_{k-1}}) = P(\omega_{i_k} | \omega_{i_{k-1}}).p(\underline{x}_k | \omega_{i_k})$$

and

$$\hat{d}(\omega_{i_1}, \omega_{i_0}) \equiv P(\omega_{i_1}) p(\underline{x}_i | \omega_{i_1})$$

The equation

$$\boxed{p(X | \Omega_i) P(\Omega_i) = P(\omega_{i_1}) p(\underline{x}_1 | \omega_{i_1}) \cdot \prod_{k=2}^{N} P(\omega_{i_k} | \omega_{i_{k-1}}) p(\underline{x}_k | \omega_{i_k})}$$

can be written as

$$\hat{D} = \prod_{k=1}^{N} \hat{d}(\omega_{i_k}, \omega_{i_{k-1}}) = p(X | \Omega_i) P(\Omega_i)$$

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

The equation

$$\hat{D} = \prod_{k=1}^{N} \hat{d}(\omega_{i_k}, \omega_{i_{k-1}}) = p(X|\Omega_i)P(\Omega_i)$$

can be written as

$$\ln(\hat{D}) = \sum_{k=1}^{N} \ln \hat{d}(\omega_{i_k}, \omega_{i_{k-1}}) \equiv \sum_{k=1}^{N} d(\omega_{i_k}, \omega_{i_{k-1}}) \equiv D$$

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

The equation

$$\hat{D} = \prod_{k=1}^{N} \hat{d}(\omega_{i_k}, \omega_{i_{k-1}}) = p(X|\Omega_i)P(\Omega_i)$$

can be written as

$$\ln(\hat{D}) = \sum_{k=1}^{N} \ln \hat{d}(\omega_{i_k}, \omega_{i_{k-1}}) \quad \equiv \sum_{k=1}^{N} d(\omega_{i_k}, \omega_{i_{k-1}}) \equiv D$$

We can use Bellman's optimality principle!!!

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

- To use Bellman's theorem
  - Define the cost up to node *k* as

$$D(\omega_{i_k}) = \sum_{r=1}^{k} d(\omega_{i_r}, \omega_{i_{r-1}})$$

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

- To use Bellman's theorem
  - Define the cost up to node *k* as

$$D(\omega_{i_k}) = \sum_{r=1}^{k} d(\omega_{i_r}, \omega_{i_{r-1}})$$

- Bellman's principle now states

$$D_{\max}(\omega_{i_k}) = \max_{i_{k-1}} \left[D_{\max}(\omega_{i_{k-1}}) + d(\omega_{i_k}, \omega_{i_{k-1}}) \right]$$

$$i_k, i_{k-1} = 1,2,\ldots,M$$

with

$$D_{\max}(\omega_{i_0}) = 0$$

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$

- Bellman's principle now states

$$D_{\max}(\omega_{i_k}) = \max_{i_{k-1}} \left[D_{\max}(\omega_{i_{k-1}}) + d(\omega_{i_k}, \omega_{i_{k-1}}) \right]$$

$$i_k, i_{k-1} = 1, 2, \ldots, M$$

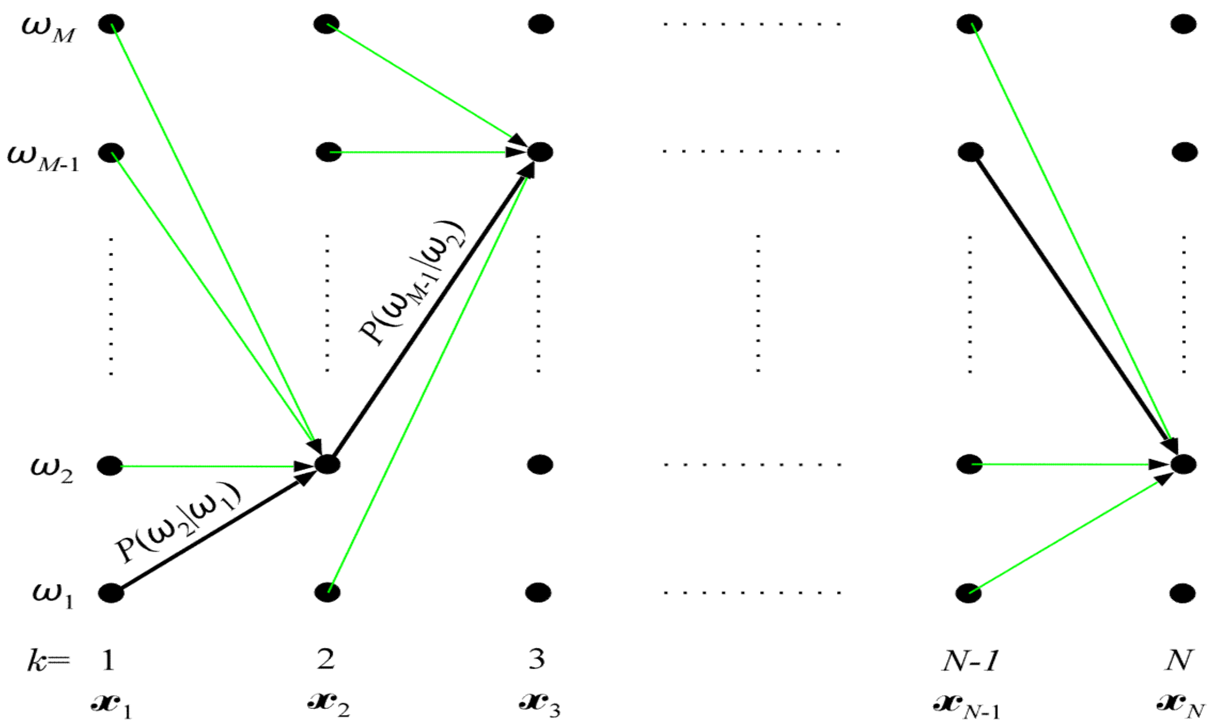
with

$$D_{\max}(\omega_{i_0}) = 0$$

- Finally, the optimal path terminates at ω_{iN}^* :

$$\omega_{i_N}^* = \arg\max_{\omega_{i_N}} D_{\max}(\omega_{i_N})$$

Viterbi Algorithm: Calculate $p(X|\Omega_i)p(\Omega_i)$



- Total M nodes in each of N columns
- Each node has M transitions
- Thus, complexity is $O\,(NM^2)$

Application in Channel Equalization

- The problem
  - Information bits I_k are transmitted
  - We receive x_k

$$I_k \rightarrow \boxed{\text{Channel}} \rightarrow x_k$$

$$x_k = f(I_k, I_{k-1}, \ldots, I_{k-n+1}) + n_k$$

where, f is the action of channel

n_k is the noise

Application in Channel Equalization

$$I_k \rightarrow \boxed{\text{Channel}} \rightarrow x_k$$

– We want the bits back

$$\underline{x}_k \rightarrow \boxed{\text{equalizer}} \rightarrow \hat{I}_k$$

where,

$$\underline{x}_k \equiv [x_k, x_{k-1}, \ldots, x_{k-l+1}]^T$$

Application in Channel Equalization

$$I_k \rightarrow \boxed{\text{Channel}} \rightarrow x_k$$

– We want the bits back

$$\underline{x}_k \rightarrow \boxed{\text{equalizer}} \rightarrow \hat{I}_{k-r}$$

OR $\underline{x}_k \rightarrow \hat{I}_k \text{ or } \hat{I}_{k-r}$ if we allow delay r

Application in Channel Equalization

$$I_k \rightarrow \boxed{\text{Channel}} \rightarrow x_k$$

– We want the bits back

$$\underline{x}_k \rightarrow \boxed{\text{equalizer}} \rightarrow \hat{I}_{k-r}$$

OR $\underline{x}_k \rightarrow \hat{I}_k \text{ or } \hat{I}_{k-r}$ if we allow delay r

This means we use the vector $\underline{x}_k$ to get a single bit $\hat{I}_k \text{ or } \hat{I}_{k-r}$

Application in Channel Equalization

- Example
  - $x_k = 0.5I_k + I_{k-1} + n_k$
  - $\underline{x}_k = \begin{bmatrix} x_k \\ x_{k-1} \end{bmatrix}, l = 2$
  - In vector $\underline{x}_k$ three input symbols are involved:

    I_k, I_{k-1}, I_{k-2}

Application in Channel Equalization

- Assuming n_k =0

I_k	I_{k-1}	I_{k-2}			$\hat{I}_k$
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Application in Channel Equalization

- Assuming n_k =0

		I_{k-2}			$\hat{I}_k$
		0			
		1			
		0			
		1			
		0			
		1			
		0			
		1			

Application in Channel Equalization

- Assuming n_k =0

	I_{k-1}	I_{k-2}			$\hat{I}_k$
	0	0			
	0	1			
	1	0			
	1	1			
	0	0			
	0	1			
	1	0			
	1	1			

Application in Channel Equalization

- Assuming n_k =0

I_k	I_{k-1}	I_{k-2}			$\hat{I}_k$
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Application in Channel Equalization

- Assuming n_k =0

	I_{k-1}	I_{k-2}		x_{k-1}	
	0	0		0	
	0	1		1	
	1	0		0.5	
	1	1		1.5	
	0	0		0	
	0	1		1	
	1	0		0.5	
	1	1		1.5	

$$I_{k-1} \rightarrow \boxed{\text{Channel}} \rightarrow x_{k-1}$$

$$x_{k-1} = 0.5 I_{k-1} + I_{k-2}$$

Application in Channel Equalization

- Assuming n_k =0

I_k	$I_{k\text{-}1}$		x_k		
0	0		0		
0	0		0		
0	1		1		
0	1		1		
1	0		0.5		
1	0		0.5		
1	1		1.5		
1	1		1.5		

$$I_k \rightarrow \boxed{\text{Channel}} \rightarrow x_k$$

$$x_k = 0.5 I_k + I_{k-1}$$

Application in Channel Equalization

- Assuming n_k =0

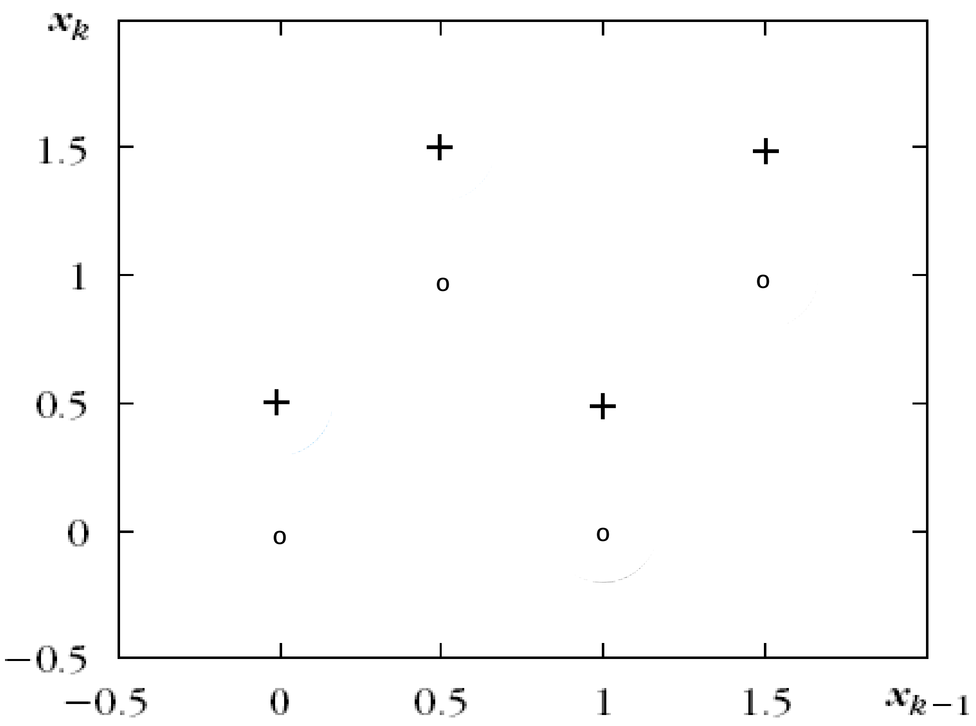
I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0	0	0	0	0	
0	0	1	0	1	
0	1	0	1	0.5	
0	1	1	1	1.5	
1	0	0	0.5	0	
1	0	1	0.5	1	
1	1	0	1.5	0.5	
1	1	1	1.5	1.5	

- Estimate I_k from (x_k, x_{k-1})

Application in Channel Equalization

- Assuming $n_k = 0$

I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0	0	0	0	0	ω_1
0	0	1	0	1	ω_2
0	1	0	1	0.5	ω_3
0	1	1	1	1.5	ω_4
1	0	0	0.5	0	ω_5
1	0	1	0.5	1	ω_6
1	1	0	1.5	0.5	ω_7
1	1	1	1.5	1.5	ω_8

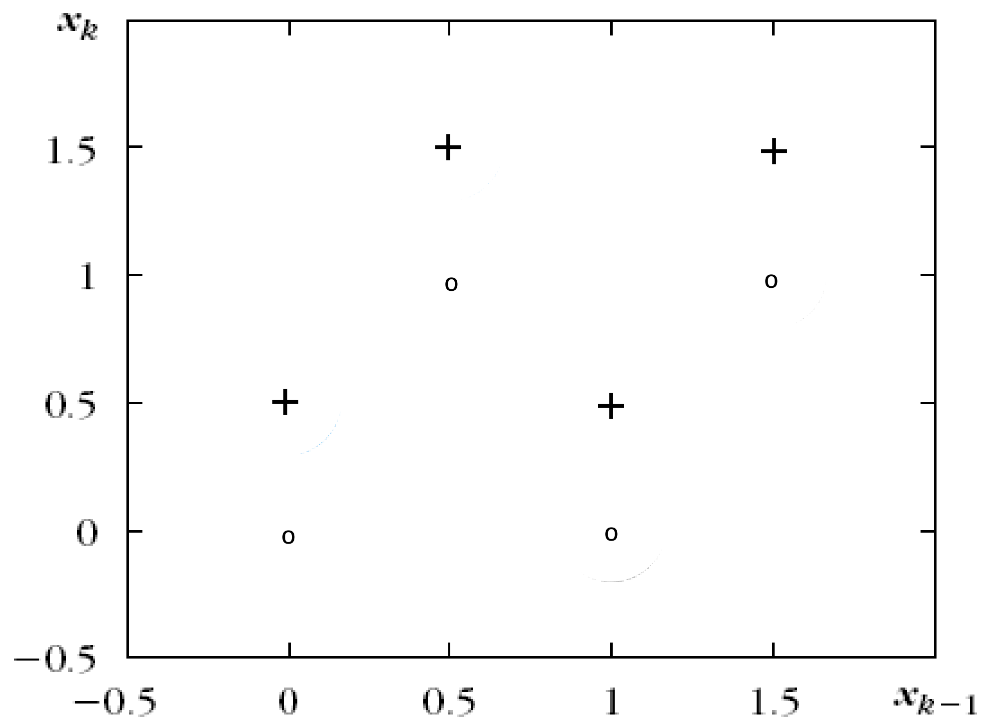


- Eight possible clusters

Application in Channel Equalization

- Assuming $n_k = 0$

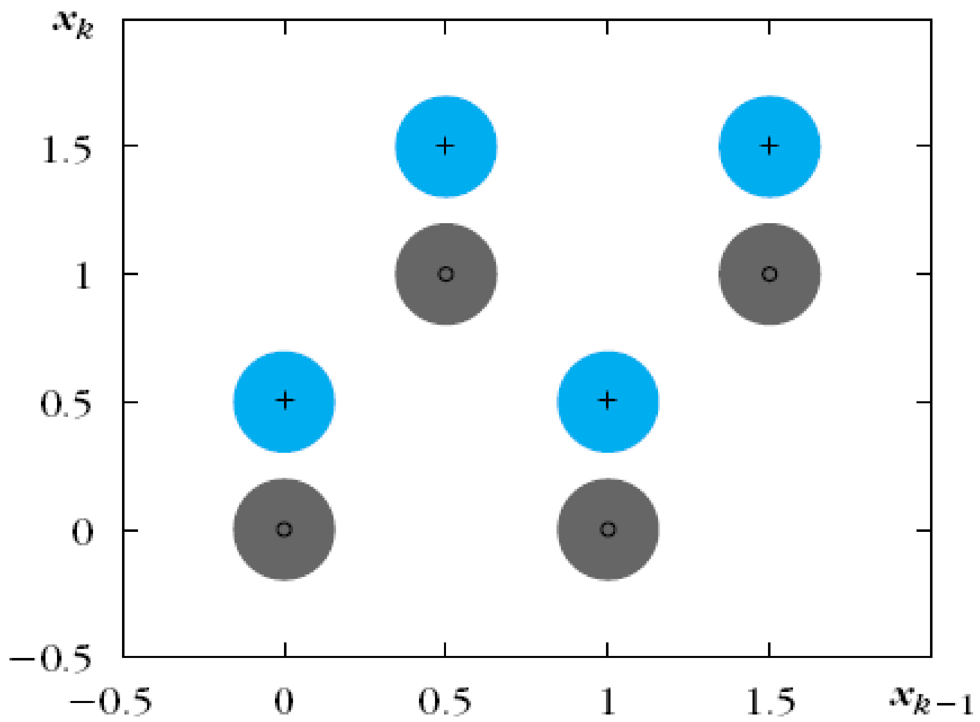
I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0	0	0	0	0	ω_1
0	0	1	0	1	ω_2
0	1	0	1	0.5	ω_3
0	1	1	1	1.5	ω_4
1	0	0	0.5	0	ω_5
1	0	1	0.5	1	ω_6
1	1	0	1.5	0.5	ω_7
1	1	1	1.5	1.5	ω_8



- '+' means I_k was 1, 'o' means I_k was 0

Application in Channel Equalization

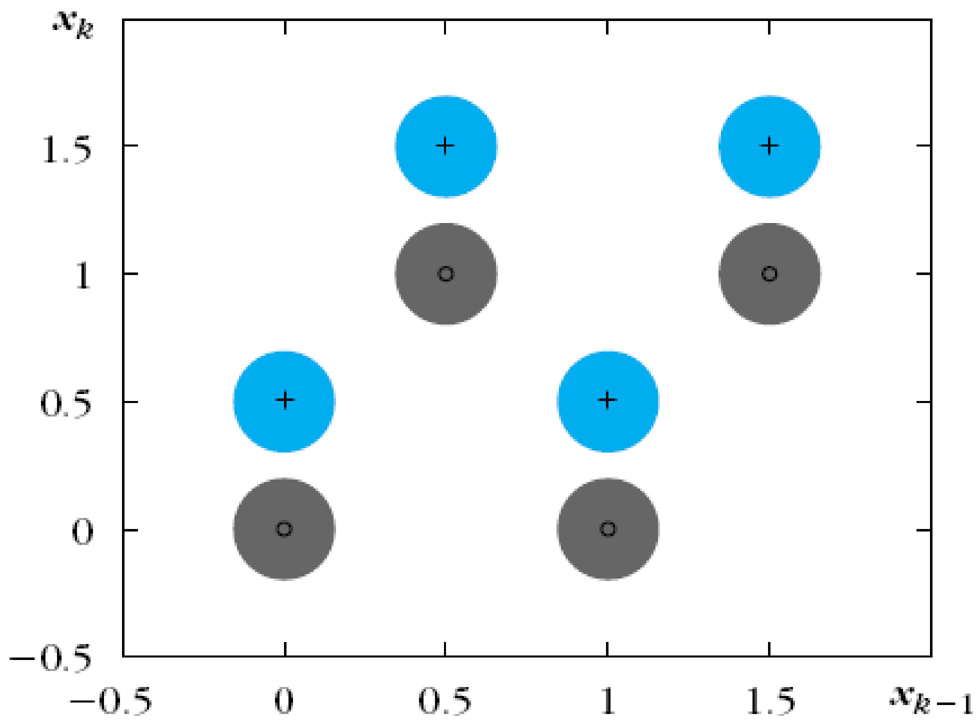
I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0	0	0	0	0	ω_1
0	0	1	0	1	ω_2
0	1	0	1	0.5	ω_3
0	1	1	1	1.5	ω_4
1	0	0	0.5	0	ω_5
1	0	1	0.5	1	ω_6
1	1	0	1.5	0.5	ω_7
1	1	1	1.5	1.5	ω_8



- Big cluster exists when the noise n_k is NOT 0 (*zero*)

Application in Channel Equalization

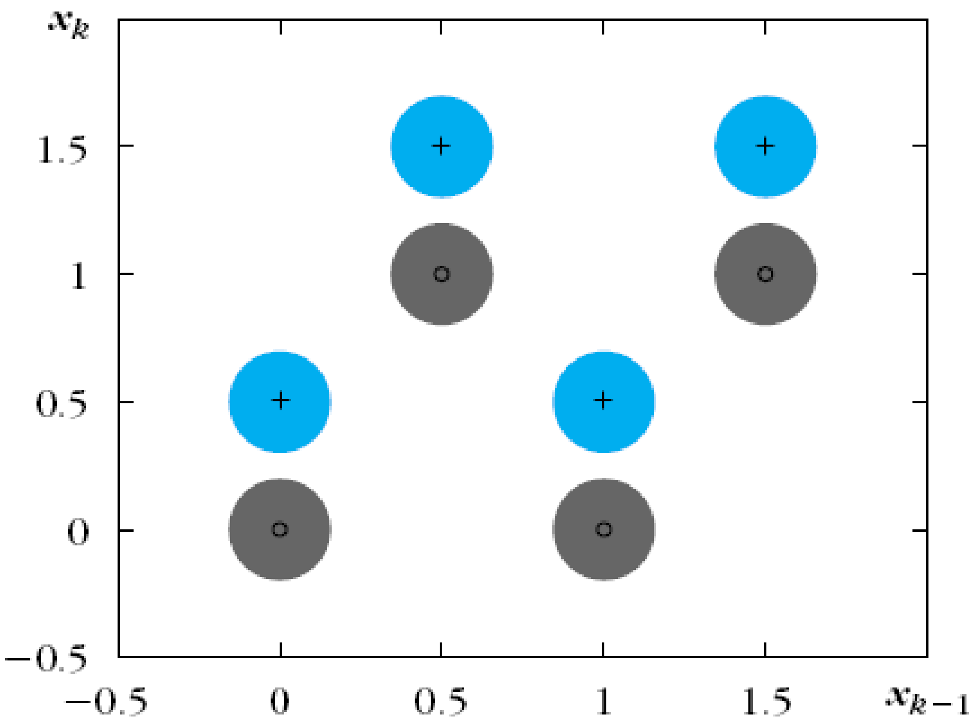
I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0	0	0	0	0	ω_1
0	0	1	0	1	ω_2
0	1	0	1	0.5	ω_3
0	1	1	1	1.5	ω_4
1	0	0	0.5	0	ω_5
1	0	1	0.5	1	ω_6
1	1	0	1.5	0.5	ω_7
1	1	1	1.5	1.5	ω_8



- Estimate $\hat{I}_k$ from (x_k, x_{k-1})
- A two class problem
- Each class is a union of clusters

Solution (1)

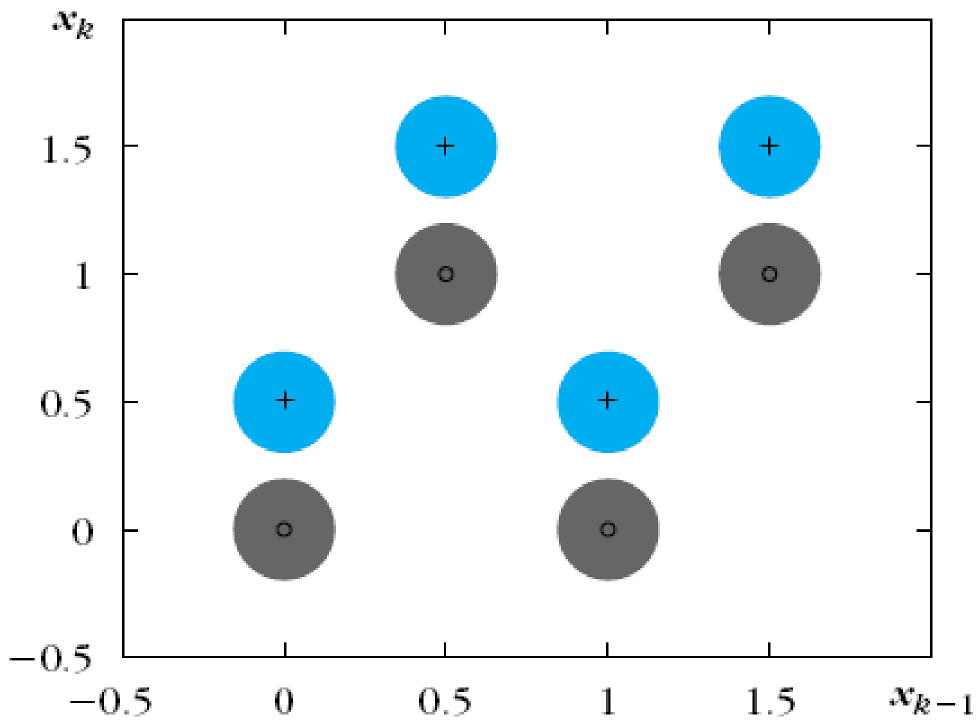
I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0	0	0	0	0	ω_1
0	0	1	0	1	ω_2
0	1	0	1	0.5	ω_3
0	1	1	1	1.5	ω_4
1	0	0	0.5	0	ω_5
1	0	1	0.5	1	ω_6
1	1	0	1.5	0.5	ω_7
1	1	1	1.5	1.5	ω_8



- During training,
  - send all values of (I_{k-2}, I_{k-1}, I_k) and calculate (x_{k-1}, x_k)
  - Calculate the cluster centers μ_k
  - Assign the bit I_k to the cluster μ_k

Solution (1)

I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
?			0	0	ω_1
?			0	1	ω_2
?			1	0.5	ω_3
?			1	1.5	ω_4
?			0.5	0	ω_5
?			0.5	1	ω_6
?			1.5	0.5	ω_7
?			1.5	1.5	ω_8



- During equalization (testing),
  - Get (x_{k-1}, x_k) from the channel
  - Find the nearest cluster μ_i
  - The bit of μ_i is the estimated transmitted bit

Solution (2)

I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0	0	0	0	0	ω_1
0	0	1	0	1	ω_2
0	1	0	1	0.5	ω_3
0	1	1	1	1.5	ω_4
1	0	0	0.5	0	ω_5
1	0	1	0.5	1	ω_6
1	1	0	1.5	0.5	ω_7
1	1	1	1.5	1.5	ω_8

- Let

$$(I_k, I_{k-1}, I_{k-2}) = (0, 0, 1)$$

which corresponds to ω_2

Solution (2)

I_k	$I_{k\text{-}1}$	$I_{k\text{-}2}$	x_k	$x_{k\text{-}1}$	
0	0	0	0	0	ω_1
0	0	1	0	1	ω_2
0	1	0	1	0.5	ω_3
0	1	1	1	1.5	ω_4
1	0	0	0.5	0	ω_5
1	0	1	0.5	1	ω_6
1	1	0	1.5	0.5	ω_7
1	1	1	1.5	1.5	ω_8

- Let

$$(I_k, I_{k-1}, I_{k-2}) = (0, 0, 1)$$

which corresponds to ω_2

Solution (2)

I_{k+1}	I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
?	0	0	0	0	0	ω_1
?	0	0	1	0	1	ω_2
?	0	1	0	1	0.5	ω_3
?	0	1	1	1	1.5	ω_4
?	1	0	0	0.5	0	ω_5
?	1	0	1	0.5	1	ω_6
?	1	1	0	1.5	0.5	ω_7
?	1	1	1	1.5	1.5	ω_8

- Let

$$(I_k, I_{k-1}, I_{k-2}) = (0, 0, 1)$$

which corresponds to ω_2

- What will be the next bit I_{k+1}

Solution (2)

I_{k+1}	I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
?	0	0	0	0	0	ω_1
0, 1	0	0	1	0	1	ω_2
?	0	1	0	1	0.5	ω_3
?	0	1	1	1	1.5	ω_4
?	1	0	0	0.5	0	ω_5
?	1	0	1	0.5	1	ω_6
?	1	1	0	1.5	0.5	ω_7
?	1	1	1	1.5	1.5	ω_8

- Let

$$(I_k, I_{k-1}, I_{k-2}) = (0, 0, 1)$$

which corresponds to ω_2

- The next bit I_{k+1} can be either 1 or 0

$$(I_{k+1}, I_k, I_{k-1})$$

(1, 0, 0) ω_5

(0, 0, 0) ω_1

Solution (2)

I_{k+1}	I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
?	0	0	0	0	0	ω_1
0, 1	0	0	1	0	1	ω_2
?	0	1	0	1	0.5	ω_3
?	0	1	1	1	1.5	ω_4
?	1	0	0	0.5	0	ω_5
?	1	0	1	0.5	1	ω_6
?	1	1	0	1.5	0.5	ω_7
?	1	1	1	1.5	1.5	ω_8

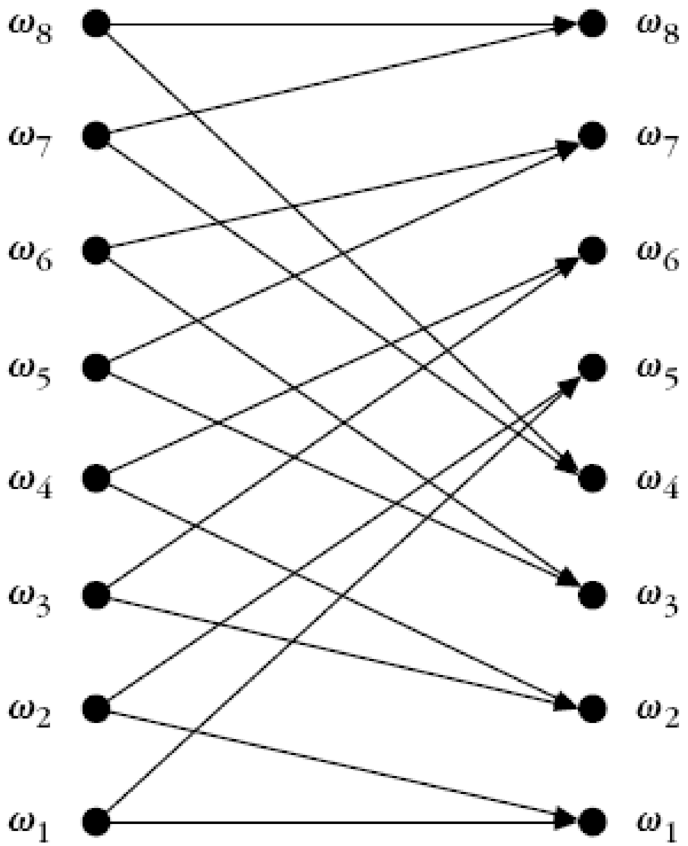
- This means, all transitions are not possible!!
- In other words, after (0, 0, 1) we will find

  either (1, 0, 0)

  or (0, 0, 0)

Solution (2)

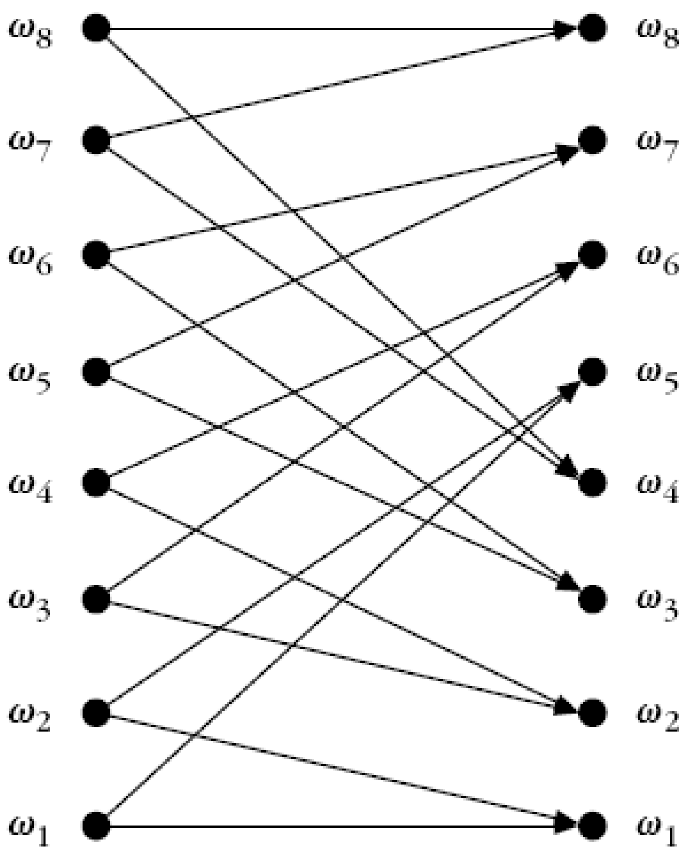
I_{k+1}	I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0, 1	0	0	0	0	0	ω_1
0, 1	0	0	1	0	1	ω_2
0, 1	0	1	0	1	0.5	ω_3
0, 1	0	1	1	1	1.5	ω_4
0, 1	1	0	0	0.5	0	ω_5
0, 1	1	0	1	0.5	1	ω_6
0, 1	1	1	0	1.5	0.5	ω_7
0, 1	1	1	1	1.5	1.5	ω_8



- This means, all transitions are NOT possible!!
- Alternately, after (0, 0, 1) we can get either (1, 0, 0) or (0, 0,0)

Solution (2)

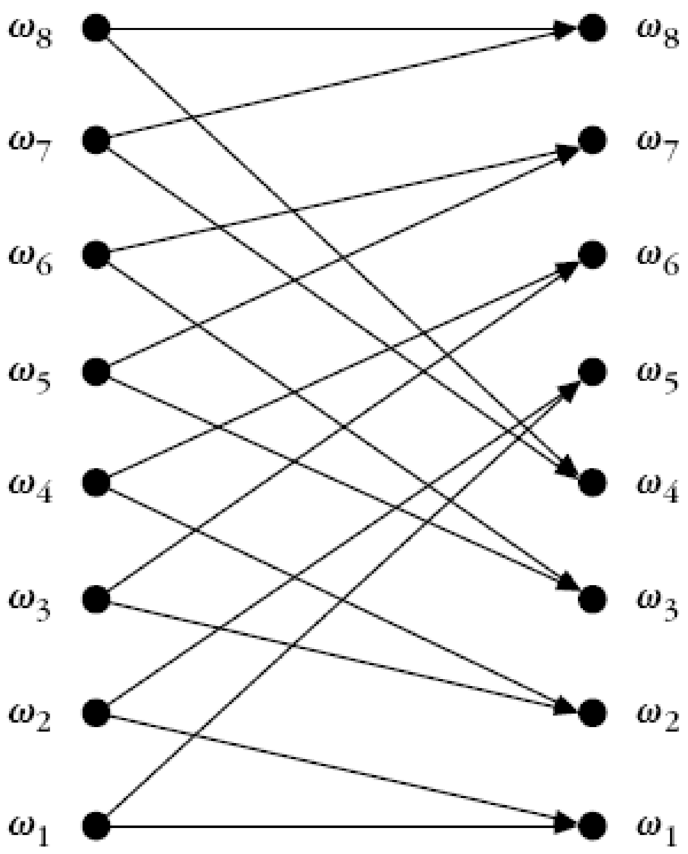
I_{k+1}	I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0, 1	0	0	0	0	0	ω_1
0, 1	0	0	1	0	1	ω_2
0, 1	0	1	0	1	0.5	ω_3
0, 1	0	1	1	1	1.5	ω_4
0, 1	1	0	0	0.5	0	ω_5
0, 1	1	0	1	0.5	1	ω_6
0, 1	1	1	0	1.5	0.5	ω_7
0, 1	1	1	1	1.5	1.5	ω_8



- We can use Viterbi algorithm now !!!

Solution (2): Viterbi Algorithm

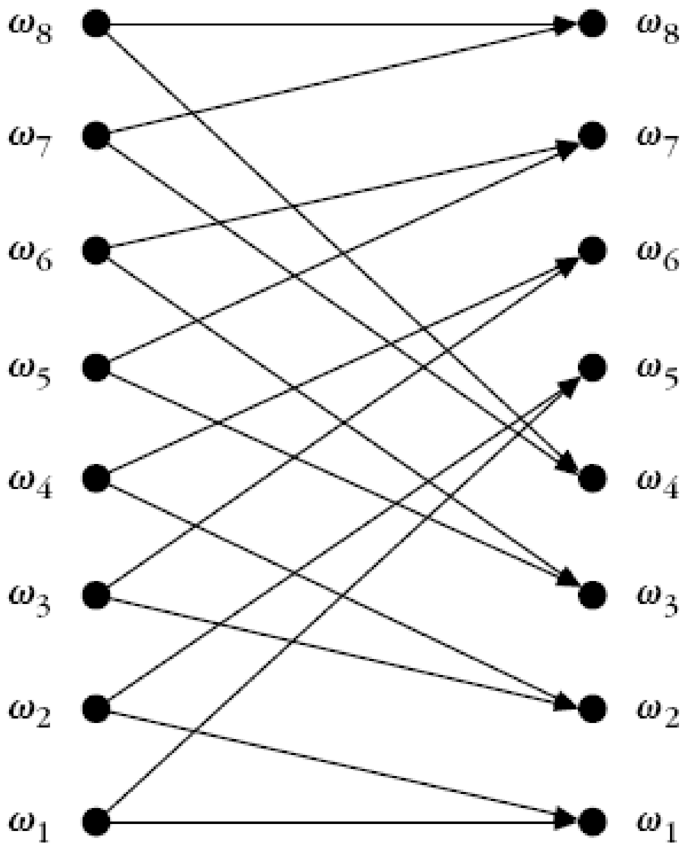
I_{k+1}	I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0, 1	0	0	0	0	0	ω_1
0, 1	0	0	1	0	1	ω_2
0, 1	0	1	0	1	0.5	ω_3
0, 1	0	1	1	1	1.5	ω_4
0, 1	1	0	0	0.5	0	ω_5
0, 1	1	0	1	0.5	1	ω_6
0, 1	1	1	0	1.5	0.5	ω_7
0, 1	1	1	1	1.5	1.5	ω_8



- We need to define:
- $P(\omega_i | \omega_j)$
- Cost function

Solution (2): Viterbi Algorithm

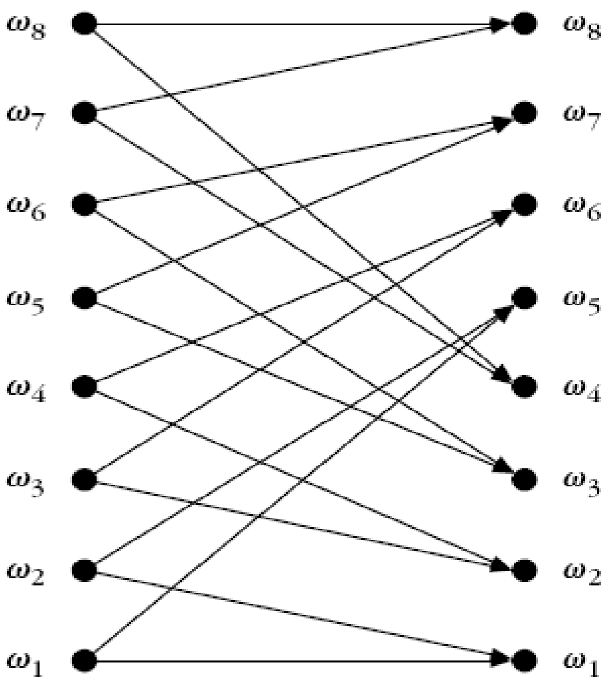
I_{k+1}	I_k	I_{k-1}	I_{k-2}	x_k	x_{k-1}	
0, 1	0	0	0	0	0	ω_1
0, 1	0	0	1	0	1	ω_2
0, 1	0	1	0	1	0.5	ω_3
0, 1	0	1	1	1	1.5	ω_4
0, 1	1	0	0	0.5	0	ω_5
0, 1	1	0	1	0.5	1	ω_6
0, 1	1	1	0	1.5	0.5	ω_7
0, 1	1	1	1	1.5	1.5	ω_8



- Assuming equiprobability of the next bit:
  - $P(\omega_1 | \omega_1) = 0.5 = P(\omega_1 | \omega_5)$

Solution (2): Viterbi Algorithm

- Cost function of transition



$$d(\omega_{i_k}, \omega_{i_{k-1}}) = d_{\omega_{i_k}}(\boldsymbol{x}_k)$$

Where, either Euclidean distance

$$d_{\omega_{i_k}}(\boldsymbol{x}_k) = \|\boldsymbol{x}_k - \boldsymbol{\mu}_{i_k}\|$$

Or Mahalanobis distance

$$d_{\omega_{i_k}}(\boldsymbol{x}_k) = \left((\boldsymbol{x}_k - \boldsymbol{\mu}_{i_k})^T \Sigma_{i_k}^{-1} (\boldsymbol{x}_k - \boldsymbol{\mu}_{i_k})\right)^{1/2}$$

Can be used