

# L23: hidden Markov models

**Discrete Markov processes**

**Hidden Markov models**

**Forward and Backward procedures**

**The Viterbi algorithm**

This lecture is based on [Rabiner and Juang, 1993]

# Introduction

## The next two lectures in the course deal with the recognition of temporal or sequential patterns

- Sequential pattern recognition is a relevant problem in several disciplines
  - Human-computer interaction: Speech recognition
  - Bioengineering: ECG and EEG analysis
  - Robotics: mobile robot navigation
  - Bioinformatics: DNA base sequence alignment

## A number of approaches can be used to perform time series analysis

- **Tap delay lines** can be used to form a feature vector that captures the behavior of the signal during a fixed time window
  - This represents a form of “short-term” memory
  - This simple approach is, however, limited by the finite length of the delay line
- **Feedback connections** can be used to produce recurrent MLP models
  - Global feedback allows the model to have “long-term” memory capabilities
  - Training and using recurrent networks is, however, rather involved and outside the scope of this class (refer to [Principe et al., 2000; Haykin, 1999])
- Instead, we will focus on **hidden Markov models**, a statistical approach that has become the “gold standard” for time series analysis

# Discrete Markov Processes

## Consider a system described by the following process

- At any given time, the system can be in one of  $N$  possible states  
 $S = \{S_1, S_2 \dots S_N\}$
- At regular times, the system undergoes a transition to a new state
- Transition between states can be described probabilistically

## Markov property

- In general, the probability that the system is in state  $q_t = S_j$  is a function of the complete history of the system
- To simplify the analysis, however, we will assume that the state of the system depends only on its immediate past

$$P(q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k \dots) = P(q_t = S_j | q_{t-1} = S_i)$$

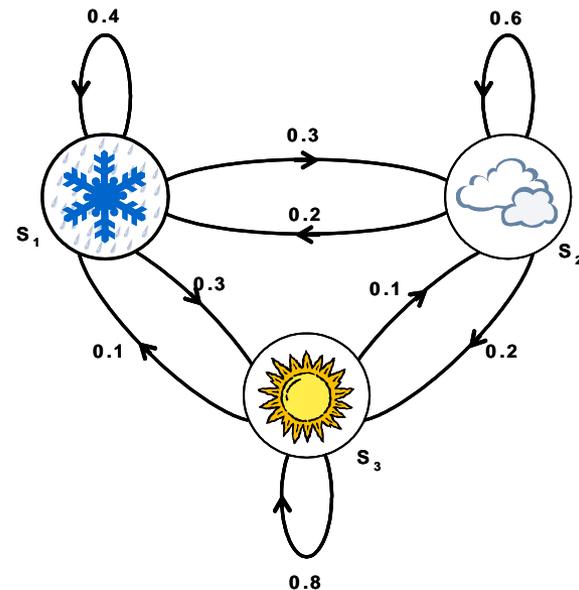
- This is known as a **first-order Markov Process**
- We will also assume that the transition probability between any two states is independent of time

$$a_{ij} = P(q_t = S_j | q_{t-1} = S_i) \text{ s.t. } \begin{cases} a_{ij} \geq 0 \\ \sum_{j=1}^N a_{ij} = 1 \end{cases}$$

## Example

- Consider a simple three-state Markov model of the weather
- Any given day, the weather can be described as being
  - State 1: precipitation (rain or snow)
  - State 2: cloudy
  - State 3: sunny
- Transitions between states are described by the transition matrix

$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$



– Question

- Given that the weather on day  $t=1$  is sunny, what is the probability that the weather for the next 7 days will be “sun, sun, rain, rain, sun, clouds, sun” ?
- Answer:

$$\begin{aligned} P(S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3 | model) \\ &= P(S_3)P(S_3|S_3)P(S_3|S_3)P(S_1|S_3)P(S_1|S_1)P(S_3|S_1)P(S_2|S_3)P(S_3|S_2) \\ &= \pi_3 a_{33} a_{33} a_{13} a_{11} a_{31} a_{23} a_{32} \\ &= 1 \times 0.8 \times 0.8 \times 0.1 \times 0.4 \times 0.3 \times 0.1 \times 0.2 \end{aligned}$$

– Question

- What is the probability that the weather stays in the same known state  $S_i$  for exactly  $T$  consecutive days?
- Answer:

$$P(q_t = S_i, q_{t+1} = S_i \dots q_{t+T} = S_{j \neq i}) = a_{ii}^{T-1}(1 - a_{ii})$$

# Hidden Markov models

## Introduction

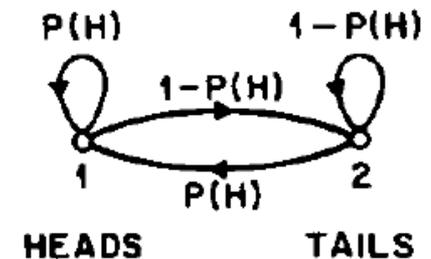
- The previous model assumes that each state can be uniquely associated with an observable event
  - Once an observation is made, the state of the system is then trivially retrieved
  - This model, however, is too restrictive to be of practical use for most realistic problems
- To make the model more flexible, we will assume that the outcomes or observations of the model are a probabilistic function of each state
  - Each state can produce a number of outputs according to a unique probability distribution, and each distinct output can potentially be generated at any state
  - These are known as *Hidden Markov Models* (HMM), because the state sequence is not directly observable, it can only be approximated from the sequence of observations produced by the system

## The coin-toss problem

- To illustrate the concept of an HMM, consider the following scenario
  - You are placed in a room with a curtain
  - Behind the curtain there is a person performing a coin-toss experiment
  - This person selects one of several coins, and tosses it: heads (H) or tails (T)
  - She tells you the outcome (H,T), but not which coin was used each time
- Your goal is to build a probabilistic model that best explains a sequence of observations  $O = \{o_1, o_2, o_3 \dots\} = \{H, T, T, H \dots\}$ 
  - The coins represent the states; these are hidden because you do not know which coin was tossed each time
  - The outcome of each toss represents an observation
  - A “likely” sequence of coins may be inferred from the observations, but this state sequence will not be unique
- If the coins are hidden, how many states should the HMM have?

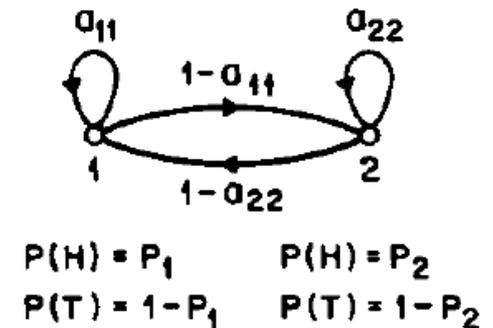
## – One-coin model

- In this case, we assume that the person behind the curtain only has one coin
- As a result, the Markov model is observable since there is only one state
- In fact, we may describe the system with a deterministic model where the states are the actual observations (see figure)
- In either case, the model parameter  $P(H)$  may be found from the ratio of heads and tails



## – Two-coin model

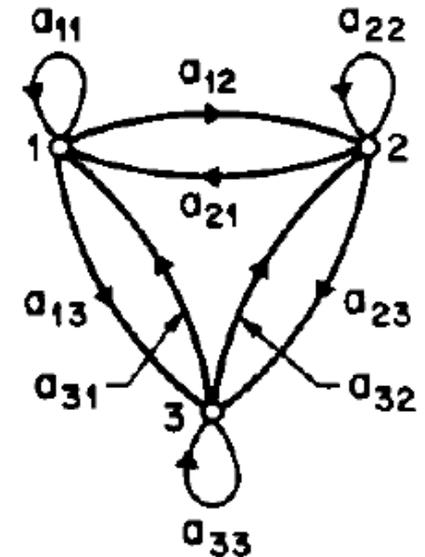
- A more sophisticated HMM would be to assume that there are two coins
  - Each coin (state) has its own distribution of heads and tails, to model the fact that the coins may be biased
  - Transitions between the two states model the random process used by the person behind the curtain to select one of the coins
- The model has 4 free parameters



[Rabiner, 1989]

– Three-coin model

- In this case, the model would have three separate states
  - This HMM can be interpreted in a similar fashion as the two-coin model
- The model has 9 free parameters



– Which of these models is best?

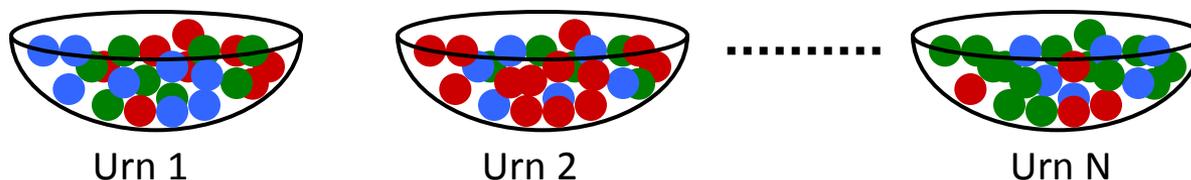
- Since the states are not observable, the best we can do is select the model that best explains the data (e.g., using a Maximum Likelihood criterion)
- Whether the observation sequence is long and rich enough to warrant a more complex model is a different story, though

	STATE		
	1	2	3
P(H)	$P_1$	$P_2$	$P_3$
P(T)	$1 - P_1$	$1 - P_2$	$1 - P_3$

[Rabiner, 1989]

## The urn-ball problem

- To further illustrate the concept of an HMM, consider this scenario
  - You are placed in the same room with a curtain
  - Behind the curtain there are  $N$  urns, each containing a large number of balls from  $M$  different colors
  - The person behind the curtain selects an urn according to an internal random process, then randomly grabs a ball from the selected urn
  - He shows you the ball, and places it back in the urn
  - This process is repeated over and over
- Questions
  - How would you represent this experiment with an HMM? What are the states? Why are the states hidden? What are the observations?



## Elements of an HMM

– An HMM is characterized by the following set of parameters

- $N$ , the number of states in the model  $S = \{S_1, S_2 \dots S_N\}$
- $M$ , the number of discrete observation symbols  $V = \{v_1, v_2 \dots v_M\}$
- $A = \{a_{ij}\}$ , the state transition probability

$$a_{ij} = P(q_{t+1} = S_j | q_t = S_i)$$

- $B = \{b_j(k)\}$ , the observation or emission probability distribution

$$b_j(k) = P(o_t = v_k | q_t = S_j)$$

- $\pi$ , the initial state distribution

$$\pi_j = P(q_1 = S_j)$$

– Therefore, an HMM is specified by two scalars ( $N$  and  $M$ ) and three probability distributions ( $A, B$ , and  $\pi$ )

- In what follows, we will represent an HMM by the compact notation

$$\lambda = (A, B, \pi)$$

## HMM generation of observation sequences

- Given a completely specified HMM  $\lambda = (A, B, \pi)$ , how can an observation sequence  $O = \{o_1, o_2, o_3, o_4, \dots\}$  be generated?
  1. Choose an initial state  $S_1$  according to the initial state distribution  $\pi$
  2. Set  $t = 1$
  3. Generate observation  $o_t$  according to the emission probability  $b_j(k)$
  4. Move to a new state  $S_{t+1}$  according to state-transition at that state  $a_{ij}$
  5. Set  $t = t + 1$  and return to 3 until  $t \geq T$
- Example
  - Generate an observation sequence with  $T = 5$  for a coin tossing experiment with three coins and the following probabilities

	$S_1$	$S_2$	$S_3$		
$P(H)$	0.5	0.75	0.25	$A = \{a_{ij}\} = \frac{1}{3} \forall i, j$	$\pi = \{\pi_i\} = \frac{1}{3} \forall i$
$P(T)$	0.5	0.25	0.75		

# The three basic HMM problems

## – Problem 1: Probability Evaluation

- Given observation sequence  $O = \{o_1, o_2, o_3 \dots\}$  and model  $\lambda = \{A, B, \pi\}$ , how do we efficiently compute  $P(O|\lambda)$ , the likelihood of the observation sequence given the model?
  - The solution is given by the Forward and Backward procedures

## – Problem 2: Optimal State Sequence

- Given observation sequence  $O = \{o_1, o_2, o_3 \dots\}$  and model  $\lambda$ , how do we choose a state sequence  $Q = \{q_1, q_2, q_3 \dots\}$  that is optimal (i.e., best explains the data)?
  - The solution is provided by the Viterbi algorithm

## – Problem 3: Parameter Estimation

- How do we adjust the parameters of the model  $\lambda = \{A, B, \pi\}$  to maximize the likelihood  $P(O|\lambda)$ 
  - The solution is given by the Baum-Welch re-estimation procedure

# Forward and Backward procedures

## Problem 1: Probability Evaluation

- Our goal is to compute the likelihood of an observation sequence  $O = \{o_1, o_2, o_3 \dots\}$  given a particular HMM model  $\lambda = \{A, B, \pi\}$
- Computation of this probability involves enumerating every possible state sequence and evaluating the corresponding probability

$$P(O|\lambda) = \sum_{\forall Q} P(O|Q, \lambda)P(Q|\lambda)$$

- For a particular state sequence  $Q = \{q_1, q_2, q_3 \dots\}$ ,  $P(O|Q, \lambda)$  is

$$P(O|Q, \lambda) = \prod_{t=1}^T P(o_t|q_t, \lambda) = \prod_{t=1}^T b_{q_t}(o_t)$$

- The probability of the state sequence  $Q$  is

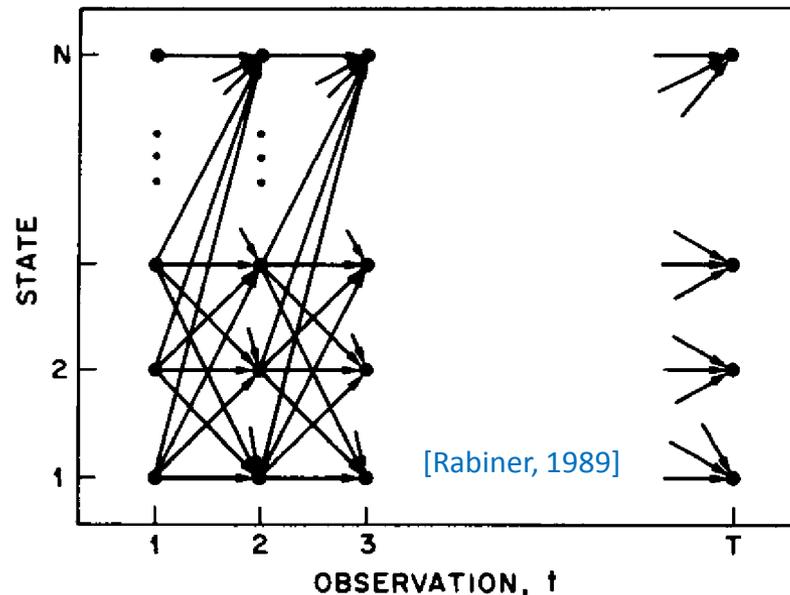
$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \dots a_{q_{T-1} q_T}$$

- Merging these results, we obtain

$$P(O|\lambda) = \sum_{q_1, q_2 \dots q_T} \pi_{q_1} b_{q_1}(o_{q_1}) a_{q_1 q_2} b_{q_2}(o_{q_2}) \dots a_{q_{T-1} q_T} b_{q_T}(o_{q_T})$$

– Computational complexity

- With  $N^T$  possible state sequences, this approach becomes unfeasible even for small problems... sound familiar?
  - For  $N = 5$  and  $T = 100$ , the order of computations is in the order of  $10^{72}$
- Fortunately, the computation of  $P(O|\lambda)$  has a lattice (or trellis) structure, which lends itself to a very efficient implementation known as the Forward procedure



## The Forward procedure

- Consider the following variable  $\alpha_t(i)$  defined as

$$\alpha_t(i) = P(o_1, o_2 \dots o_t, q_t = S_i | \lambda)$$

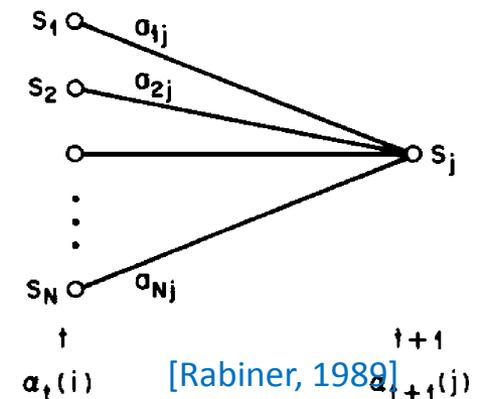
- which represents the probability of the observation sequence up to time  $t$  AND the state  $S_i$  at time  $t$ , given model  $\lambda$
- Computation of this variable can be efficiently performed by induction

- **Initialization:**  $\alpha_1(i) = \pi_i b_i(o_1)$

- **Induction:**  $\alpha_{t+1}(j) = [\sum_{i=1}^N \alpha_t(i) a_{ij}] b_j(o_{t+1}) \quad \begin{cases} 1 \leq t \leq T-1 \\ 1 \leq j \leq N \end{cases}$

- **Termination:**  $P(O|\lambda) = \sum_{i=1}^N \alpha_T(i)$

- As a result, computation of  $P(O|\lambda)$  can be reduced from  $2T \times N^T$  down to  $N^2 \times T$  operations (from  $10^{72}$  to 3000 for  $N = 5, T = 100$ )



## The Backward procedure

- Analogously, consider the backward variable  $\beta_t(i)$  defined as

$$\beta_t(i) = P(o_{t+1}, o_{t+2} \dots o_T | q_t = S_i, \lambda)$$

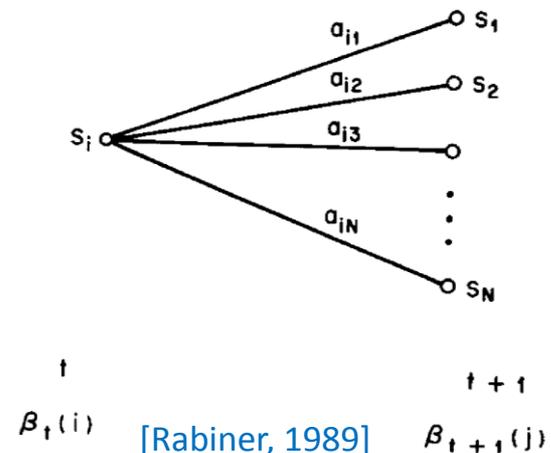
- $\beta_t(i)$  represents the probability of the partial observation sequence from  $t + 1$  to the end, given state  $S_i$  at time  $t$  and model  $\lambda$

- As before,  $\beta_t(i)$  can be computed through induction

- **Initialization:**  $\beta_T(i) = 1$  (arbitrarily)

- **Induction:**  $\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \quad \begin{cases} t = T - 1, T - 2 \dots 1 \\ 1 \leq i \leq N \end{cases}$

- Similarly, this computation can be effectively performed in the order of  $N^2 \times T$  operations



# The Viterbi algorithm

## Problem 2: Optimal State Sequence

- Finding the optimal state sequence is more difficult problem than the estimation of  $P(O|\lambda)$
- Part of the issue has to do with defining an optimality measure, since several criteria are possible
  - Finding the states  $q_t$  that are *individually* more likely at each time  $t$
  - Finding the *single best* state sequence path (i.e., maximize the posterior  $P(O|Q, \lambda)$ )
- The second criterion is the most widely used, and leads to the well-known Viterbi algorithm
  - However, we first optimize the first criterion as it allows us to define a variable that will be used later in the solution of Problem 3

- As in the Forward-Backward procedures, we define a variable  $\gamma_t(i)$

$$\gamma_t(i) = P(q_t = S_i | O, \lambda)$$

- which represents the probability of being in state  $S_i$  at time  $t$ , given the observation sequence  $O$  and model  $\lambda$

- Using the definition of conditional probability, we can write

$$\gamma_t(i) = P(q_t = S_i | O, \lambda) = \frac{P(O, q_t = S_i | \lambda)}{P(O | \lambda)} = \frac{P(O, q_t = S_i | \lambda)}{\sum_{i=1}^N P(O, q_t = S_i | \lambda)}$$

- Now, the numerator of  $\gamma_t(i)$  is equal to the product of  $\alpha_t(i)$  and  $\beta_t(i)$

$$\gamma_t(i) = \frac{P(O, q_t = S_i | \lambda)}{\sum_{i=1}^N P(O, q_t = S_i | \lambda)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^N \alpha_t(i)\beta_t(i)}$$

- The individually most likely state  $q_t^*$  at each time is then

$$q_t^* = \arg \max_{1 \leq i \leq N} [\gamma_t(i)] \quad \forall t = 1 \dots T$$

- The problem with choosing the individually most likely states is that the overall state sequence may not be valid
  - Consider a situation where the individually most likely states are  $q_t = S_i$  and  $q_{t+1} = S_j$ , but the transition probability  $a_{ij} = 0$
- Instead, and to avoid this problem, it is common to look for the *single best* state sequence, at the expense of having sub-optimal individual states
- This is accomplished with the Viterbi algorithm

## The Viterbi algorithm

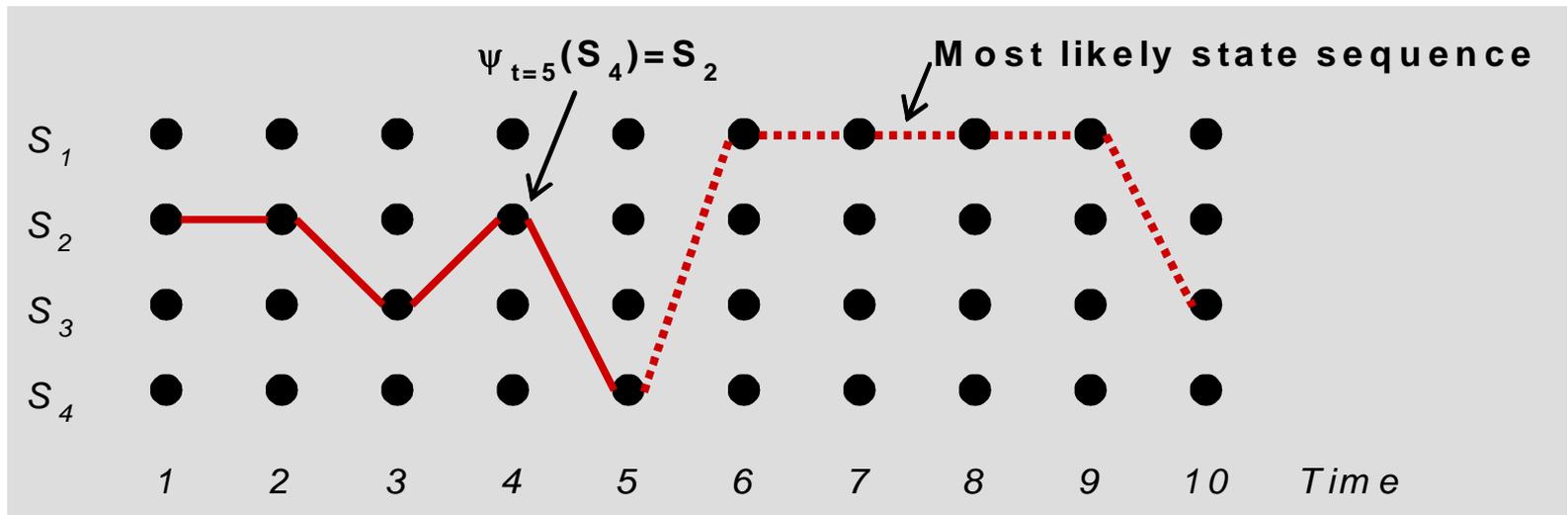
- To find the single best state sequence we define yet another variable

$$\delta_t(i) = \max_{q_1 q_2 \dots q_{t-1}} P(q_1 q_2 \dots q_t = S_i, o_1 o_2 \dots o_t | \lambda)$$

- which represents the highest probability along a single path that accounts for the first  $t$  observations and ends at state  $S_i$
- By induction,  $\delta_{t+1}(j)$  can be computed as
$$\delta_{t+1}(j) = \max_i [\delta_t(i) a_{ij}] b_j(o_{t+1})$$
- To retrieve the state sequence, we also need to keep track of the state that maximizes  $\delta_t(i)$  at each time  $t$ , which is done by constructing an array

$$\Psi_{t+1}(j) = \arg \max_{1 \leq i \leq N} [\delta_t(i) a_{ij}]$$

- $\Psi_{t+1}(j)$  is the state at time  $t$  from which a transition to state  $S_j$  maximizes the probability  $\delta_{t+1}(j)$



– The Viterbi algorithm for finding the optimal state sequence becomes

• **Initialization:** 
$$\begin{cases} \delta_1(i) = \pi_i b_i(o_1) & 1 \leq i \leq N \\ \Psi_1(i) = 0 & \text{(no previous states)} \end{cases}$$

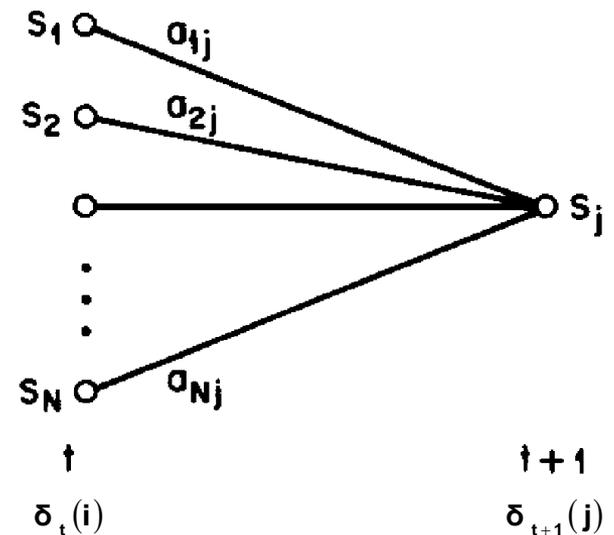
• **Recursion:** 
$$\left. \begin{aligned} \delta_t(j) &= \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] b_j(o_t) \\ \Psi_t(j) &= \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}] \end{aligned} \right\} 2 \leq t \leq T; 1 \leq j \leq N$$

• **Termination:** 
$$\begin{cases} P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \\ q_T^* = \arg \max_{1 \leq i \leq N} [\delta_T(i)] \end{cases}$$

– And the optimal state sequence can be retrieved by backtracking

$$q_t^* = \Psi_{t+1}(q_{t+1}^*) \quad t = T - 1, T - 2 \dots 1$$

– Notice that the Viterbi algorithm is similar to the Forward procedure, except that it uses a maximization over previous states instead of a summation



[Rabiner, 1989]