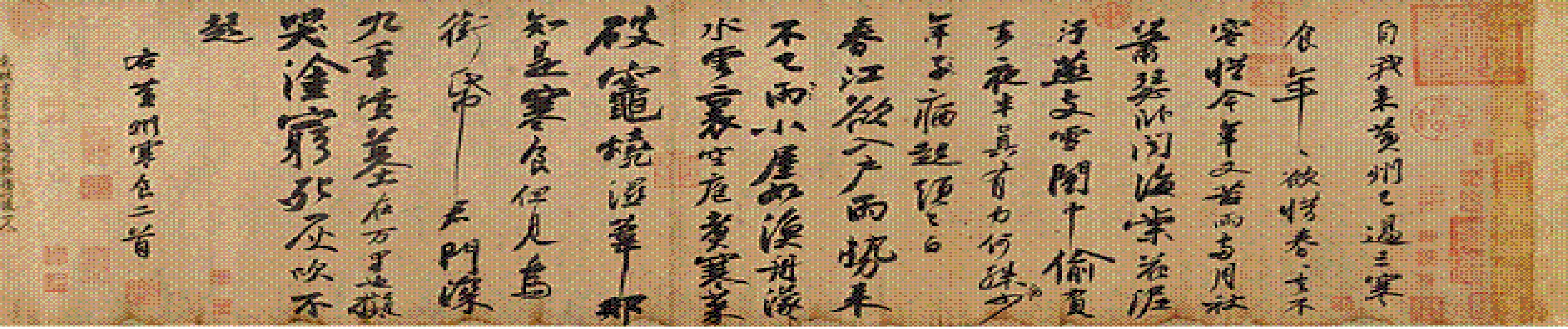


计算语言学
Computational Linguistics

教师：孙茂松

Tel:62781286

Email:sms@tsinghua.edu.cn

TA：**林衍凯**

Email:linyankai423@qq.com

郑重声明

- 此课件仅供选修清华大学计算机系研究生课《计算语言学》(70240052)的学生个人学习使用，所以只允许学生将其下载、存贮在自己的电脑中。未经孙茂松本人同意，任何人不得以任何方式扩散之（包括不得放到任何服务器上）。否则，由此可能引起的一切涉及知识产权的法律责任，概由该人负责。
- 此课件仅限孙茂松本人讲课使用。除孙茂松本人外，凡授课过程中，PTT文件显示此《郑重声明》之情形，即为侵权使用。

第六章
词性自动标注
（隐Markov模型）

6.1. Part-of-Speech Tagging

Task: Select the most likely sequence of syntactic categories for the words in a sentence.

The simplest algorithm: to disambiguate words based solely on the largest probability that a word occurs with a particular tag.

90% success rate

Consider context! Use of some of the local context of the sentence in which the word appears

'flies' prefers V PROB(0.0005) than N PROB(0.0003) if without context

But given the prior word is 'the', 'flies' will prefer N

6.1. Part-of-Speech Tagging

Solution?

Let $w_1,\ldots\ldots,w_T$ be a sequence of words(sentence). The goal is to find the sequence of lexical categories $C_1,\ldots\ldots,C_T$ that maximizes

$$PROB(C_1,\ldots\ldots,C_T \mid w_1,\ldots\ldots,w_T) \qquad \text{(formula 1)}$$

According to Bayes' rule:

$$PROB(C_1,\ldots\ldots,C_T \mid w_1,\ldots\ldots,w_T)$$

$$= \frac{PROB(C_1,\ldots\ldots,C_T) \times PROB(w_1\ldots\ldots w_T \mid C_1,\ldots\ldots C_T)}{PROB(w_1,\ldots\ldots,w_T)} \qquad \text{(formula 2)}$$

Thus the problem reduces to finding the sequence $C_1,\ldots\ldots,C_T$ that maximizes

$$PROB(C_1,\ldots\ldots,C_T) \times PROB(w_1,\ldots\ldots,w_T \mid C_1,\ldots\ldots C_T) \qquad \text{(formula 3)}$$

Approximation: making some independence assumptions.

6.1. Part-of-Speech Tagging

▲ **Approximation for the first expression in formula 3**

Using the definition of conditional probability sequentially:

$$PROB(C_1,......,C_T)$$

$$= PROB(C_1,......,C_{T-1}) \times PROB(C_T \mid C_1,......,C_{T-1})$$

$$= PROB(C_1,......,C_{T-2}) \times PROB(C_{T-1} \mid C_1,......,C_{T-2}) \times PROB(C_T \mid C_1,......,C_{T-1})$$

$$= ......$$

$$= PROB(C_1 C_2) \times PROB(C_3 \mid C_1 C_2) \times ...... \times PROB(C_{T-1} \mid C_1,......,C_{T-2}) \times PROB(C_T \mid C_1,......,C_{T-1})$$

$$= PROB(C_1) \times PROB(C_2 \mid C_1) \times PROB(C_3 \mid C_1 C_2) \times ...... \times PROB(C_{T-1} \mid C_1,......,C_{T-2}) \times PROB(C_T \mid C_1,......,C_{T-1})$$

n-gram models:

bigram $PROB(C_i \mid C_{i-1})$

trigram $PROB(C_i \mid C_{i-1} C_{i-2})$

Using bigram approximation:

$$PROB(C_1,......,C_T) \cong PROB(C_1 \mid C_0) \times PROB(C_2 \mid C_1) \times ...... \times PROB(C_T \mid C_{T-1})$$

$$= \prod_{i=1,T} PROB(C_i \mid C_{i-1})$$

6.1. Part-of-Speech Tagging

To account the beginning of a sentence: pseudo-category ϕ at position 0 as the value of C_0

Example:

$$
\begin{aligned}
&PROB(ART\ \ N\ \ V\ \ N) \\
&= PROB(ART \mid \phi) \times PROB(N \mid ART) \times PROB(V \mid N) \times PROB(N \mid V)
\end{aligned}
$$

▲ **Approximation for the second expression in formula 3**

$$PROB(w_1,\ldots\ldots,w_T \mid C_1,\ldots\ldots C_T) \cong \prod_{i=1,T} PROB(w_i \mid C_i)$$

Example:

$$
\begin{aligned}
&PROB(the\ fly\ likes\ flowers \mid ART\ \ N\ \ V\ \ N) \\
&= PROB(the \mid ART) \times PROB(fly \mid N) \times PROB(likes \mid V) \times PROB(flowers \mid N)
\end{aligned}
$$

6.1. Part-of-Speech Tagging

▲ **The combination of the above two approximations**

With these two approximations, the problem changes into finding the sequence $C_1, \ldots\ldots, C_T$ that maximizes the value of

$$\prod_{i=1,T} PROB(C_i \mid C_{i-1}) \times \prod_{i=1,T} PROB(w_i \mid C_i)$$

$$= \prod_{i=1,T} PROB(C_i \mid C_{i-1}) \times PROB(w_i \mid C_i) \qquad \text{(formula 4)}$$

▲ **Train the statistical parameters by annotated corpus**

bigram probabilities:

$$PROB(C_i = V \mid C_{i-1} = N) \cong \frac{Count(N \ \ at \ \ position \ \ i-1 \ \ and \ \ V \ \ at \ \ i)}{Count(N \ \ at \ \ position \ \ i-1)}$$

lexical-generation probabilities:

$$PROB(w_i \mid C_i) \cong \frac{Count(w_i \ \ with \ \ C_i)}{Count(C_i)}$$

6.1. Part-of-Speech Tagging

$$
\begin{aligned}
& PROB(N\ V\ ART\ N) \\
&= PROB(N \mid \phi) \times PROB(V \mid N) \times PROB(ART \mid V) \times PROB(N \mid ART) \\
&= 0.29 \times 0.43 \times 0.65 \times 1 = 0.081
\end{aligned}
$$

$$
\begin{aligned}
& PROB(flies\ likes\ a\ flower \mid N\ \ V\ \ ART\ N) \\
&= PROB(flies \mid N) \times PROB(likes \mid V) \times PROB(a \mid ART) \times PROB(flower \mid N) \\
&= 0.025 \times 0.1 \times 0.36 \times 0.063 = 0.000054 = 5.4 \times 10^{-5}
\end{aligned}
$$

=>HMM (Hidden Markov Model)

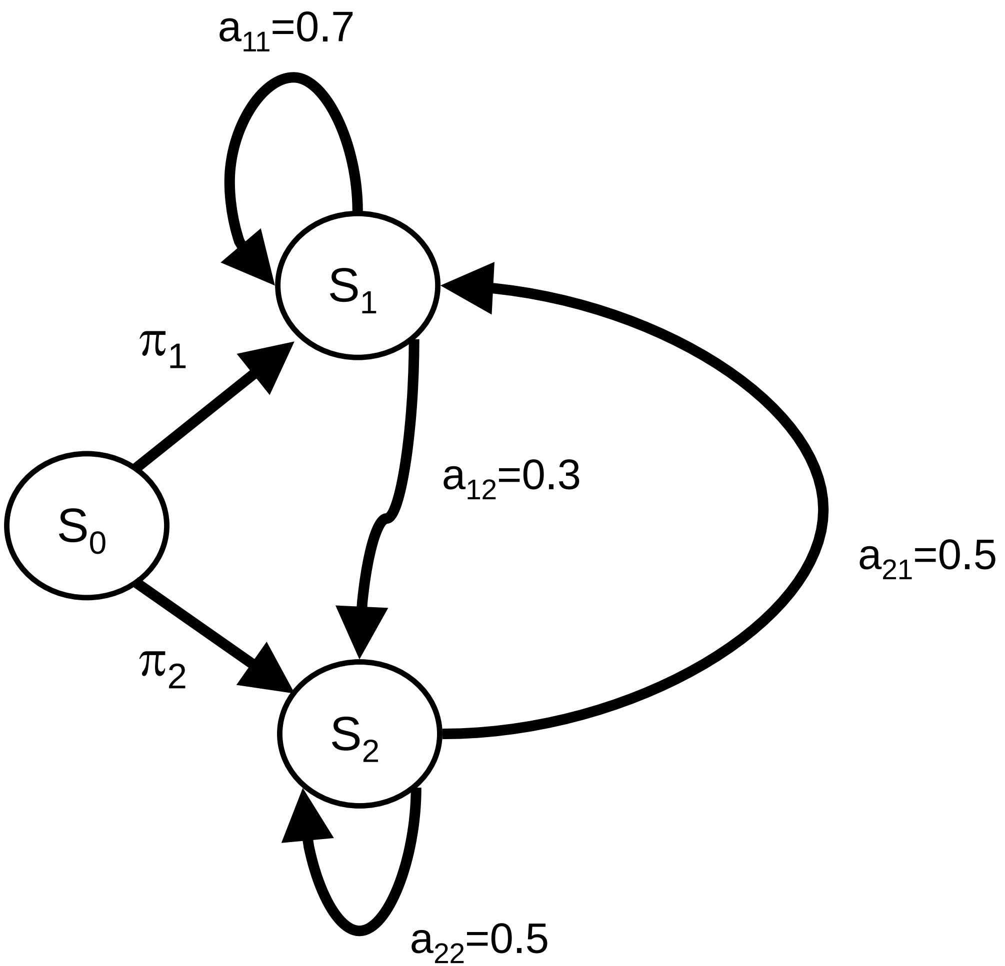
Solution? **To find the sequence** $C_1, \ldots\ldots, C_T$ **that maximizes the value of**

$$
\begin{aligned}
& \prod_{i=1,T} PROB(C_i \mid C_{i-1}) \times \prod_{i=1,T} PROB(w_i \mid C_i) \\
&= \prod_{i=1,T} PROB(C_i \mid C_{i-1}) \times PROB(w_i \mid C_i)
\end{aligned}
$$

Brute force method: combinatorial explosion problem

6.1. Part-of-Speech Tagging

VMM (Visible Markov Model)



6.1. Part-of-Speech Tagging

HMM

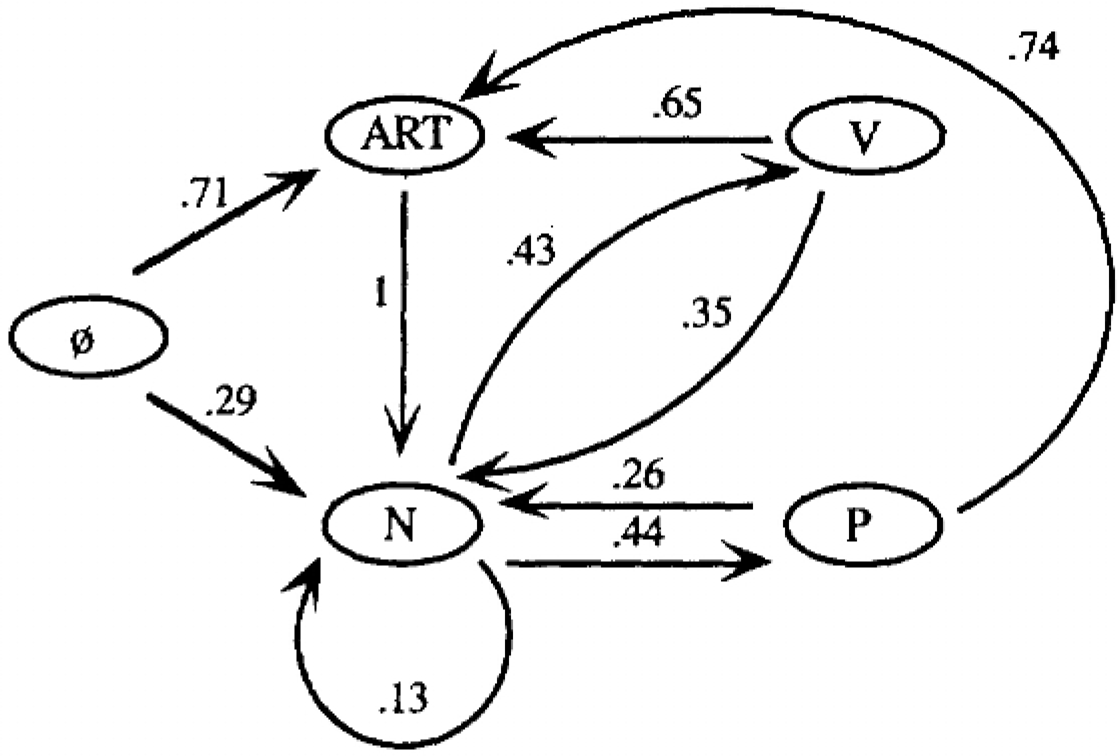
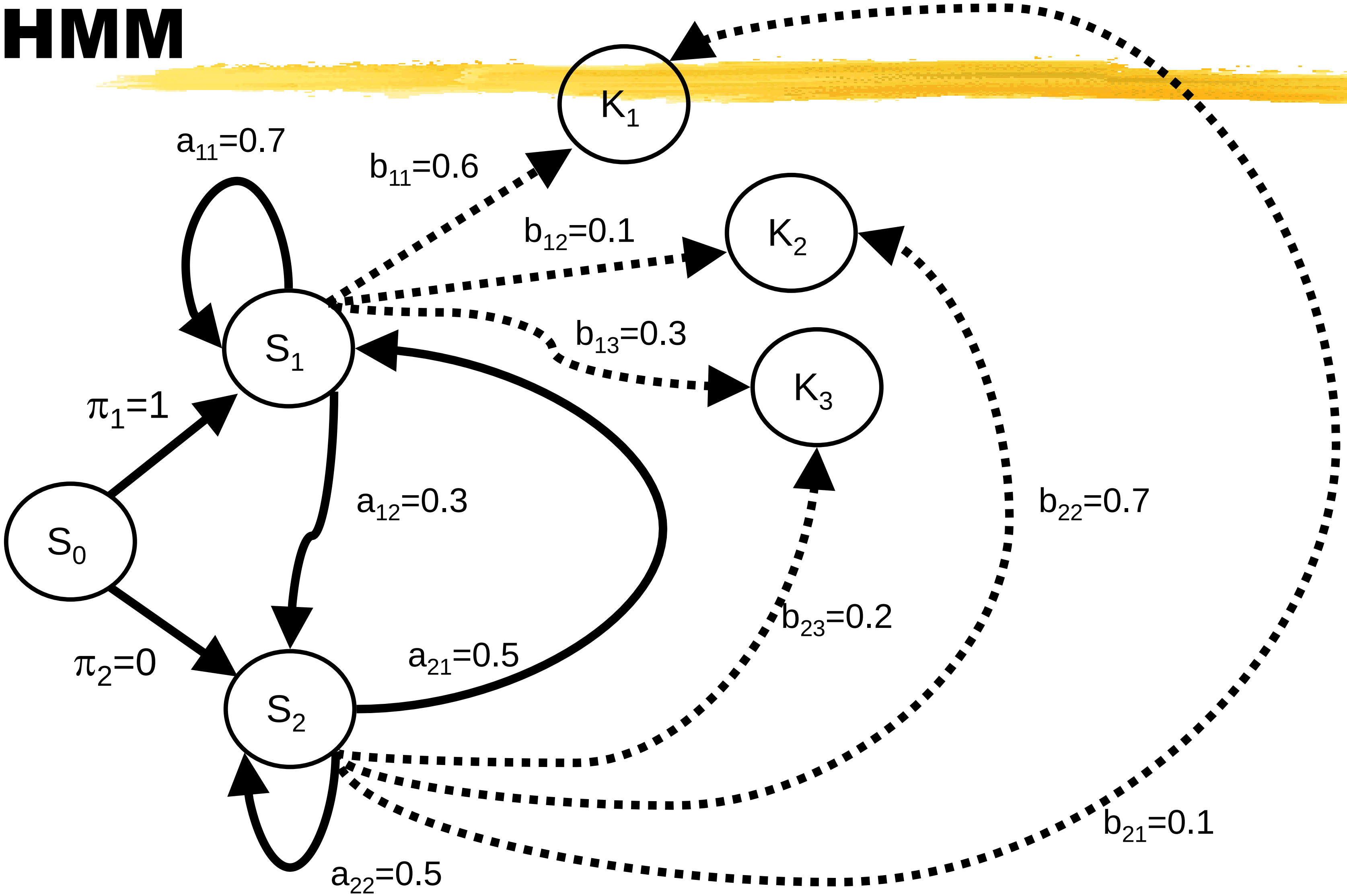


Figure 7.7 A Markov chain capturing the bigram probabilities

6.1. Part-of-Speech Tagging

HMM



6.1. Part-of-Speech Tagging

The Viterbi Algorithm

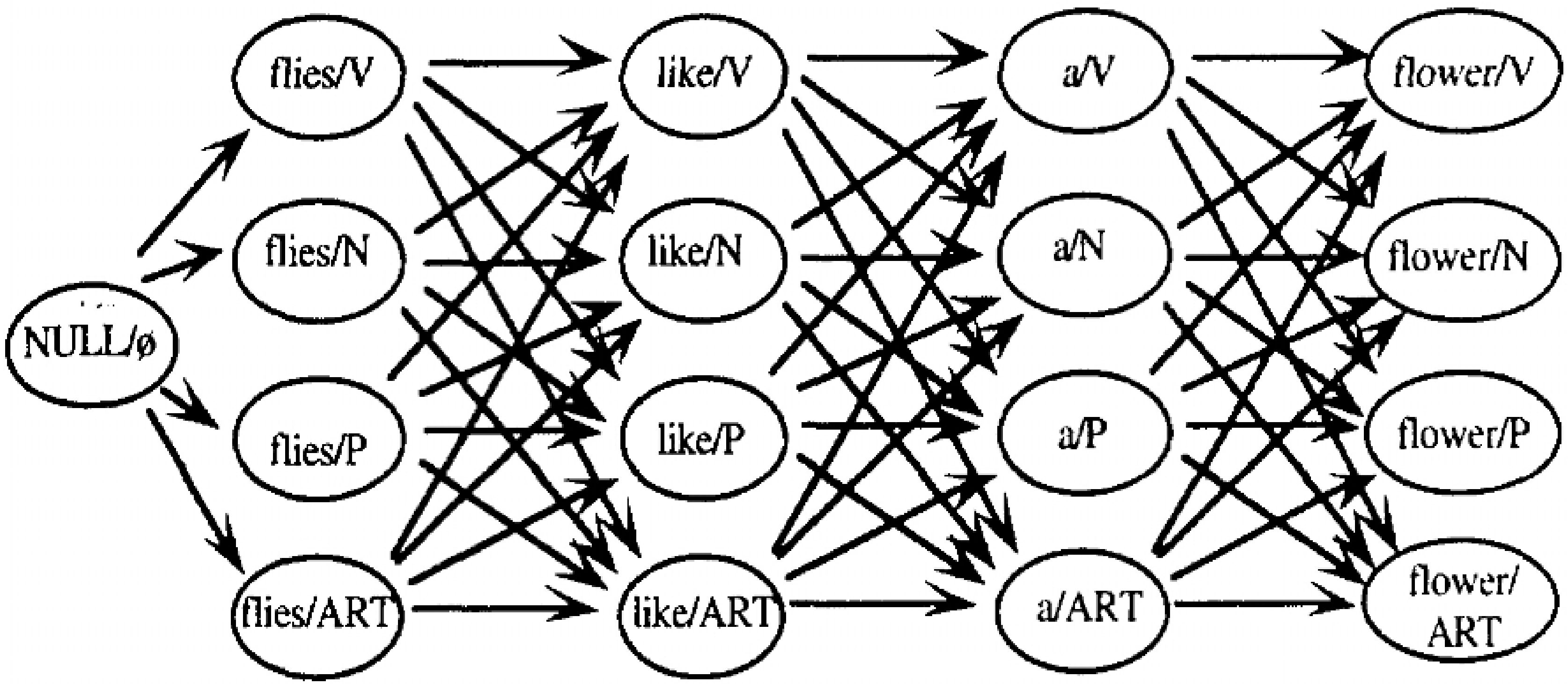


Figure 7.8 Encoding the 256 possible sequences exploiting the Markov assumption

6.1. Part-of-Speech Tagging

The Viterbi Algorithm

Data structure used in the algorithm:

▲ A $N \times T$ array SEQSCORE(n,t): record the probability for the best sequence up to the position t that ends with a word in category L_n, where N is the number of lexical categories (L_1,……,L_N) and T is the number of words in the sentence (w_1,……,w_T).

SEQSCORE	1	2	……	T
1				
……				
N				

▲ A $N \times T$ array BACKPTR: indicate for each category at each position t what the preceding category is in the best sequence at position t-1.

BACKPTR	1	2	……	T
1				
……				
N				

6.1. Part-of-Speech Tagging

The Viterbi algorithm:

Given word sequence $w_1, \ldots, w_T$, lexical categories $L_1, \ldots, L_N$, lexical probabilities $PROB(w_t \mid L_i)$, and bigram probabilities $PROB(L_i \mid L_j)$, find the most likely sequence of lexical categories $C_1, \ldots, C_T$ for the word sequence.

Initialization Step

For i = 1 to N do

$SEQSCORE(i,1) = PROB(w_1 \mid L_i) * PROB(L_i \mid \phi)$

$BACKPTR(i,1) = 0$

Iteration Step

For t = 2 to T do

For i = 1 to N do

$SEQSCORE(i,t) =$

$MAX_{j=1,..N}(SEQSCORE(j,t-1) * PROB(L_i \mid L_j)) * PROB(w_t \mid L_i)$

$BACKPTR(i,t)$ = index of j that gave the max above

Sequence Identification Step

C(T) = i that maximizes $SEQSCORE(i, T)$

For i = T-1 to 1 do C(i) = $BACKPTR$(C(i+1),i+1)

6.1. Part-of-Speech Tagging

Example: *Flies like a flower.*

Bigram probabilities:

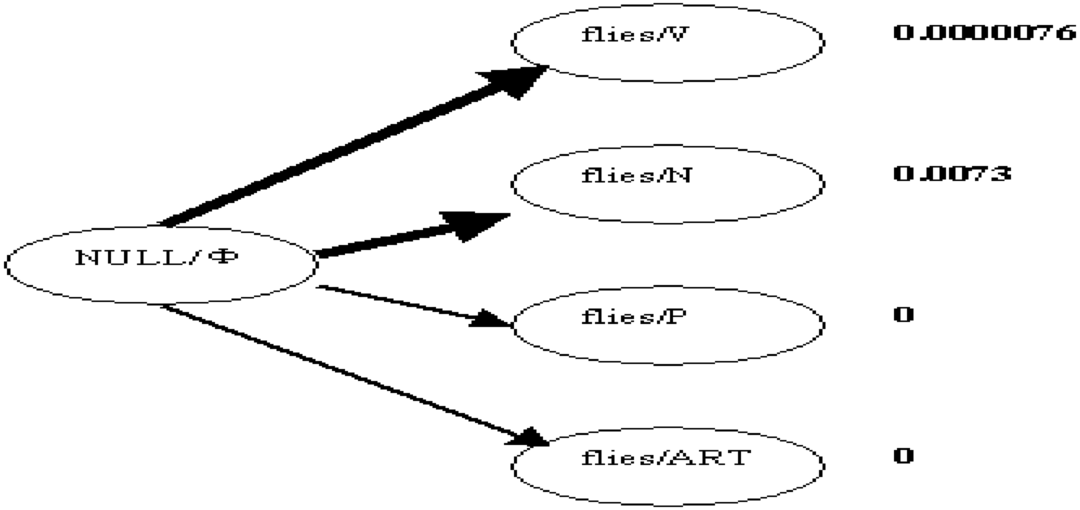
$PROB(ART \mid \phi) = 0.71$	$PROB(N \mid V) = 0.35$
$PROB(N \mid \phi) = 0.29$	$PROB(ART \mid V) = 0.65$
$PROB(N \mid ART) = 1$	$PROB(ART \mid P) = 0.74$
$PROB(V \mid N) = 0.43$	$PROB(N \mid P) = 0.26$
$PROB(N \mid N) = 0.13$	$PROB(other\ bigram) = 0.0001$
$PROB(P \mid N) = 0.44$	

The lexical-generation probabilities:

$PROB(flies \mid N) = 0.025$	$PROB(a \mid ART) = 0.36$
$PROB(flies \mid V) = 0.076$	$PROB(a \mid N) = 0.001$
$PROB(like \mid V) = 0.10$	$PROB(flower \mid N) = 0.063$
$PROB(like \mid P) = 0.068$	$PROB(flower \mid V) = 0.05$
$PROB(like \mid N) = 0.012$	

6.1. Part-of-Speech Tagging

(1) Initialization Step



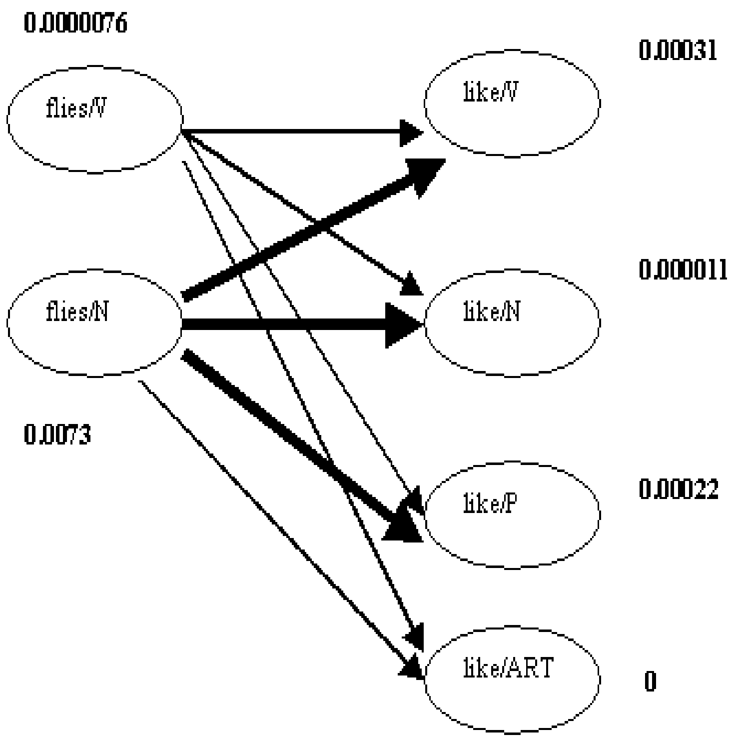
SEQSCORE	1(flies)	2(like)	3(a)	4(flower)
1(V)	0.0000076			
2(N)	0.0073			
3(P)	0			
4(ART)	0			

BACKPTR	1(flies)	2(like)	3(a)	4(flower)
1(V)	0			
2(N)	0			
3(P)	0			
4(ART)	0			

6.1. Part-of-Speech Tagging

(2) Iteration Step

(2.1) t=2

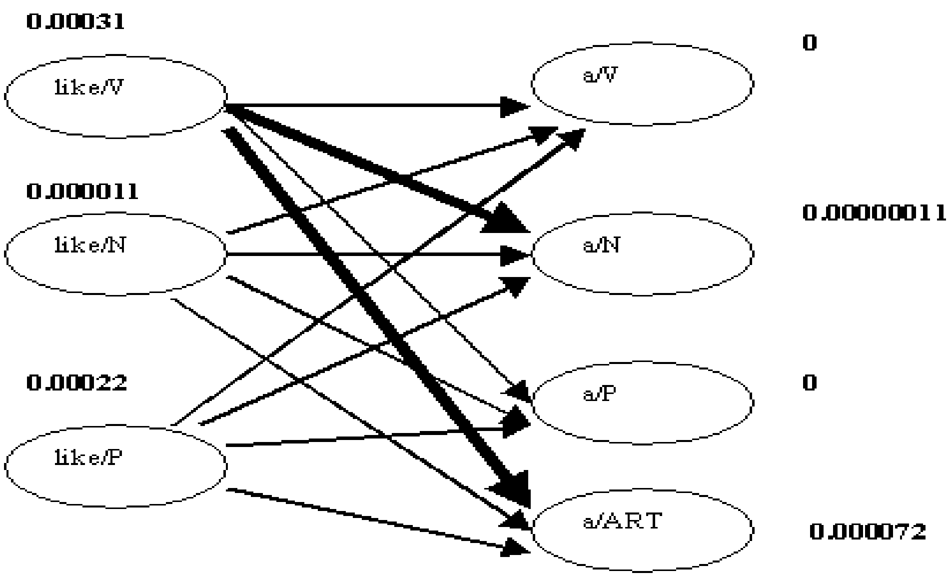


SEQSCORE	1(flies)	2(like)	3(a)	4(flower)
1(V)	0.0000076	0.00031		
2(N)	0.0073	0.000011		
3(P)	0	0.00022		
4(ART)	0	0		

BACKPTR	1(flies)	2(like)	3(a)	4(flower)
1(V)	0	2		
2(N)	0	2		
3(P)	0	2		
4(ART)	0			

6.1. Part-of-Speech Tagging

(2.2) t=3

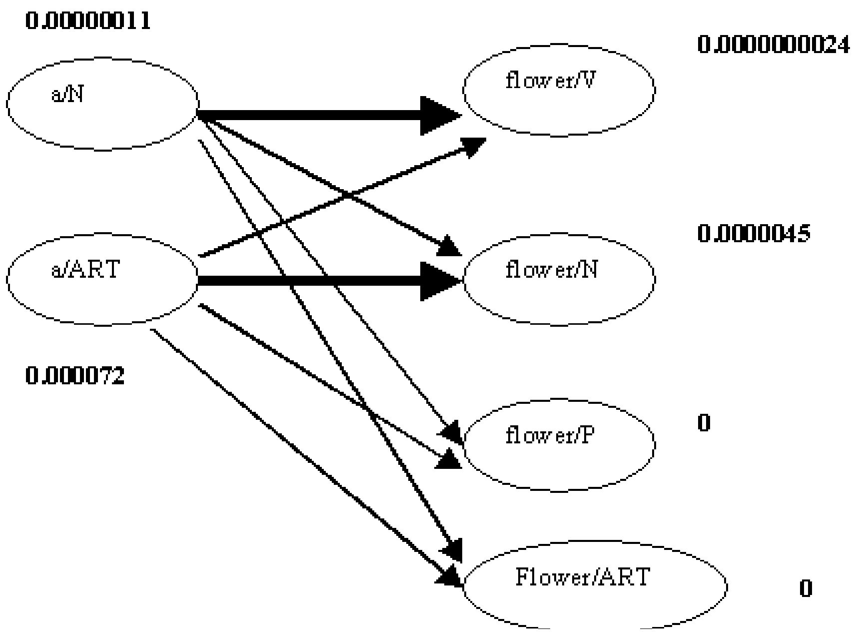


SEQSCORE	1(flies)	2(like)	3(a)	4(flower)
1(V)	0.0000076	0.00031	0	
2(N)	0.0073	0.000011	0.00000011	
3(P)	0	0.00022	0	
4(ART)	0	0	0.000072	

BACKPTR	1(flies)	2(like)	3(a)	4(flower)
1(V)	0	2		
2(N)	0	2	1	
3(P)	0	2		
4(ART)	0		1	

6.1. Part-of-Speech Tagging

(2.3) t=4



SEQSCORE	1(flies)	2(like)	3(a)	4(flower)
1(V)	0.0000076	0.00031	0	0.0000000024
2(N)	0.00725	0.000011	0.00000011	0.0000045
3(P)	0	0.00022	0	0
4(ART)	0	0	0.000072	0

BACKPTR	1(flies)	2(like)	3(a)	4(flower)
1(V)	0	2		2
2(N)	0	2	1	4
3(P)	0	2		
4(ART)	0		1	

6.1. Part-of-Speech Tagging

(3) Sequence Identification Step

C(4)=2

i=3: C(3)=BACKPTR(C(4), 4)= BACKPTR(2, 4)=4

i=2: C(2)=BACKPTR(C(3), 3)= BACKPTR(4, 3)=1

i=1: C(1)=BACKPTR(C(2), 2)= BACKPTR(1, 2)=2

Solution: flies(N) like(V) a(ART) flower(N)

6.2. Tag Set

1.	CC	Coordinating conjunction	19.	PP$	Possessive pronoun
2.	CD	Cardinal number	20.	RB	Adverb
3.	DI'	Determiner	21.	RBR	Comparative adverb
4.	EX	Existential *there*	22.	RBS	Superlative Adverb
5.	FW	Foreign word	23.	RP	Particle
6.	IN	Preposition / subord. conj	24.	SYM	Symbol (math or scientific)
7.	11	Adjective	25.	10	to
8.	JJR	Comparative adjective	26.	UH	Interjection
9.	JJS	Superlative adjective	27.	VB	Verb, base form
10.	LS	List item marker	28.	VBD	Verb, past tense
11.	MD	Modal	29.	VBG	Verb, gerund/pres. participle
12.	NN	Noun, singular or mass	30.	VBN	Verb, past participle
13.	NNS	Noun, plural	31.	VBP	Verb, non-3s, present
14.	NNP	Proper noun, singular	32.	VBZ	Verb, 3s, present
15.	NNPS	Proper noun, plural	33.	WDT	Wh-determiner
16.	PDT	Predeterminer	34.	WP	Wh-pronoun
17.	POS	Possessive ending	35.	WPZ	Possessive vh-pronoun
18.	PRP	Personal pronoun	36.	WRB	Wh-adverb

The Penn Treebank tag set

6.3. Related Factors

(1) Handling Unknown Words:

$\mathbf{w_1 \quad w_2 \quad w_3}$

$\mathbf{C_1 \quad C_2}$

pick the category **C** for the unknown word that maximizes

$\mathbf{PROB(C \mid C_1 \; C_2)}$

(2) n-grams? n=3? ……

(3) 中文：分词与词性标注一体化？

6.4. The Three Fundamental Questions for HMM

General form of an HMM:

Set of states	$S = \{s_1, \ldots s_N\}$
Output alphabet	$K = \{k_1, \ldots, k_M\} = \{1, \ldots, M\}$
Intial state probabilities	$\Pi = \{\pi_i\}, i \in S$
State transition probabilities	$A = \{a_{ij}\}, i, j \in S$
Symbol emission probabilities	$B = \{b_{ijk}\}, i, j \in S, k \in K$
State sequence	$X = (X_1, \ldots, X_{T+1}) \quad X_t : S \mapsto \{1, \ldots, N\}$
Output sequence	$O = (o_1, \ldots, o_T) \quad o_t \in K$

$t := 1;$

Start in state s_i with probability π_i (i.e., $X_1 = i$)

forever do

  Move from state s_i to state s_j with probability a_{ij} (i.e., $X_{t+1} = j$)

  Emit observation symbol o_t = k with probability b_{ijk}

  $t := t + 1$

end

6.4. The Three Fundamental Questions for HMM

The Three Fundamental Questions for HMMs

There are three fundamental questions that we want to know about an HMM:

1. Given a model $\mu = (A, B, \Pi)$, how do we efficiently compute how likely a certain observation is, that is $P(O|\mu)$?

2. Given the observation sequence 0 and a model μ, how do we choose a state sequence $(X_1, \ldots, X_{T+1})$ that best explains the observations?

3. Given an observation sequence 0, and a space of possible models found by varying the model parameters $\mu = (A, B, \pi)$, how do we find the model that best explains the observed data?

Fundamental question 1:

Finding the probability of an observation

Given the observation sequence $O = (o_1, \ldots, o_T)$ and a model $\mu = (A, B, \Pi)$, *we* wish to know how to efficiently compute $P(O|\mu)$ – the probability of the observation given the model. This process is often referred to as *decoding*.

For any state sequence $X = (X1, \ldots, X_{T+1})$,

$$P(O|X,\mu) = \prod_{t=1}^{T} P(o_t|X_t, X_{t+1}, \mu)$$
$$= b_{X_1X_2o_1} b_{X_2X_3o_2} \ldots b_{X_TX_{T+1}o_T}$$

and,

$$P(X|\mu) = \pi_{X_1} a_{X_1X_2} a_{X_2X_3} \cdots a_{X_TX_{T-1}}$$

Now,

$$P(O, X|\mu) = P(O|X, \mu) P(X|\mu)$$

$(2T+1) \cdot N^{T+1}$ multiplications

Therefore,

$$P(O|\mu) = \sum_{X} P(O|X, \mu) P(X|\mu)$$
$$= \sum_{X_1 \cdots X_{T+1}} \pi_{X_1} \prod_{t=1}^{T} a_{X_tX_{t+1}} b_{X_tX_{t+1}o_t}$$

The forward procedure

The form of caching that is indicated in these diagrams is called the forward *procedure.* We describe it in terms of forward variables:

$\alpha_i(t) = P(o_1 o_2 \ldots o_{t-1}, X_t = i|\mu)$

The forward variable α_i (t) is stored at (s_i, t) in the trellis and expresses the total probability of ending up in state s_i at time t (given that the observations $o_1 \cdots o_{t-1}$ were seen). It is calculated by summing probabilities for all incoming arcs at a trellis node. We calculate the forward variables in the trellis left to right using the following procedure:

1. Initialization

$$\alpha_i(1) = \pi_i, \quad 1 \leq i \leq N$$

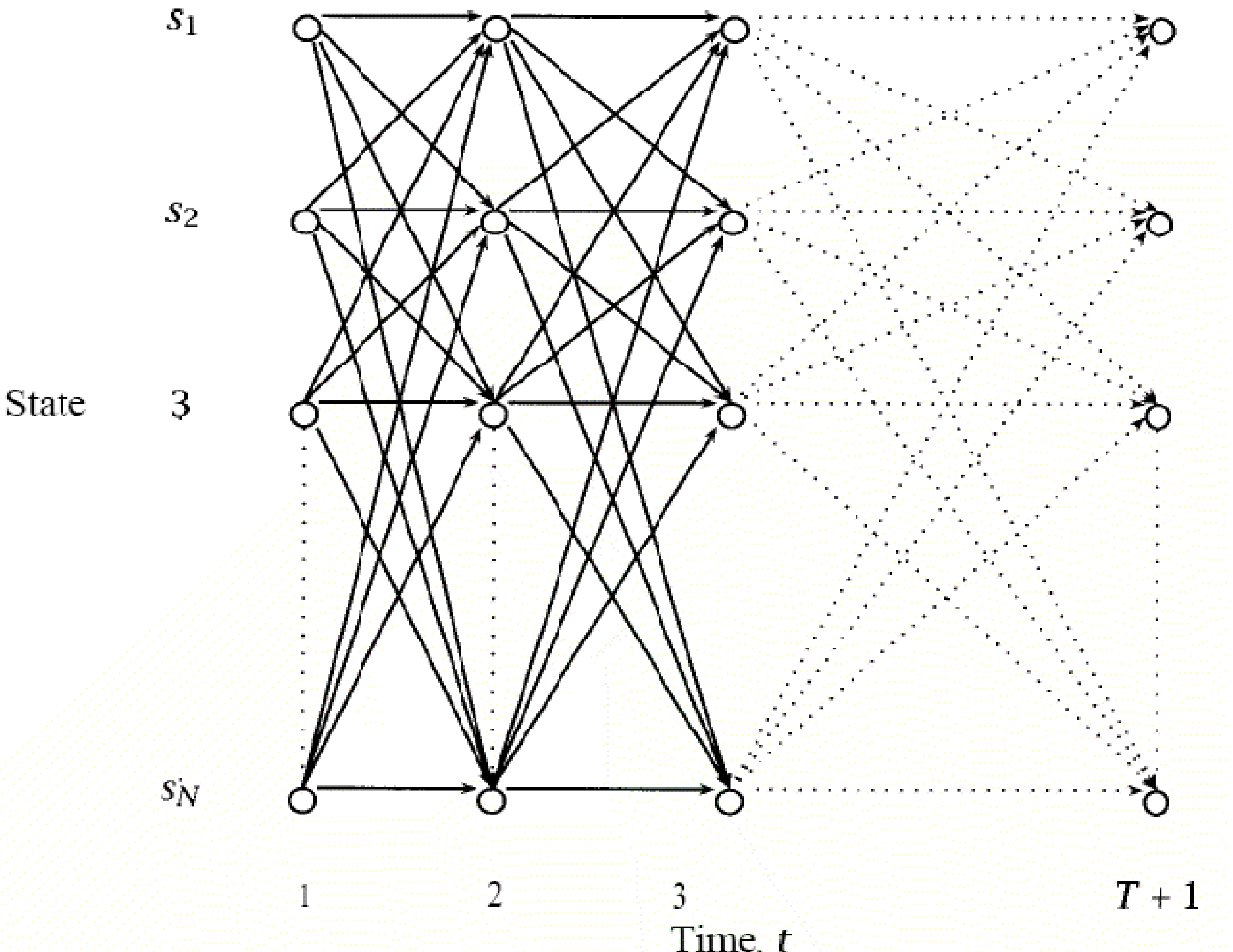
2. Induction

$$\alpha_j(t+1) = \sum_{i=1}^{N} \alpha_i(t) a_{ij} b_{ijo_t}, \quad 1 \leq t \leq T, 1 \leq j \leq N$$

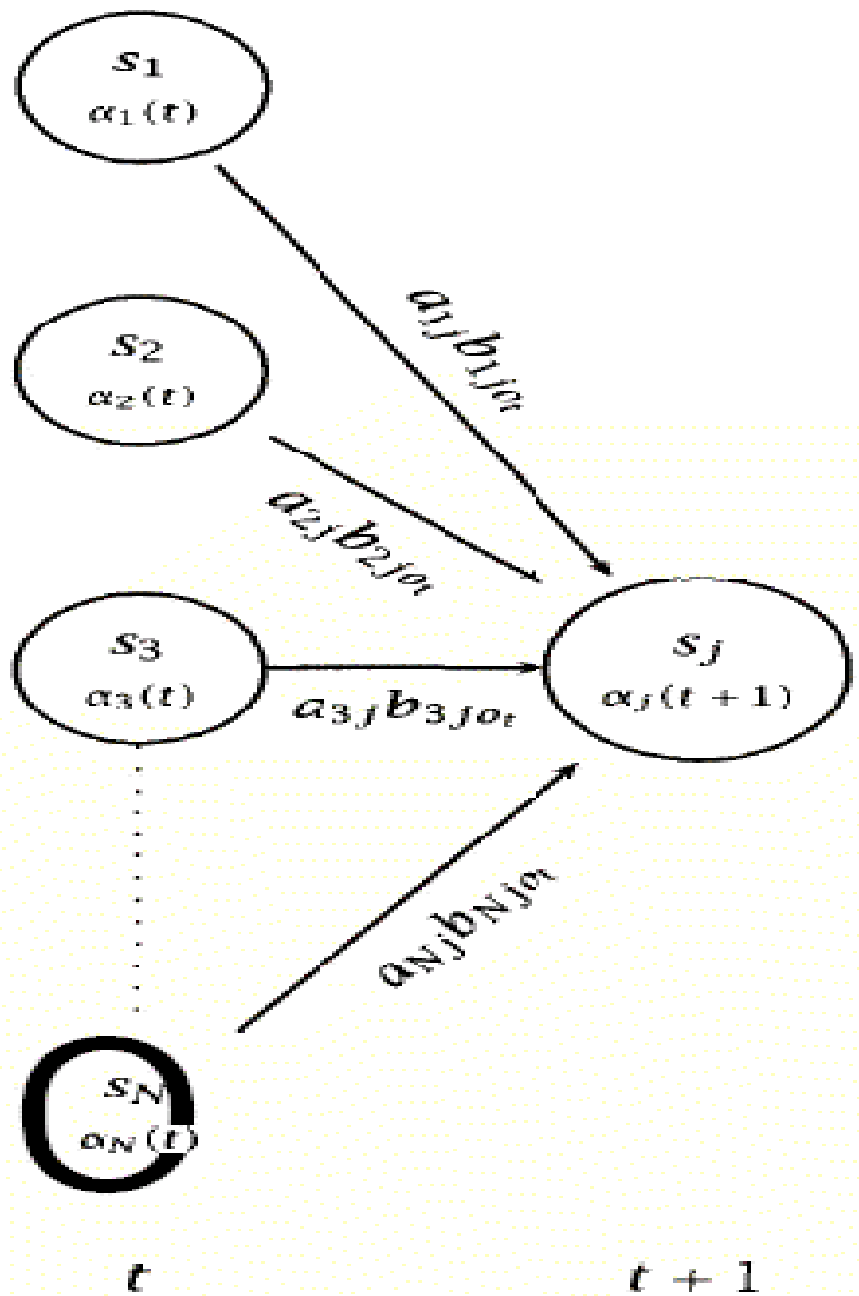
3. Total

$$P(O|\mu) = \sum_{i=1}^{N} \alpha_i(T+1)$$

This is a much cheaper algorithm that requires only $2N^2T$ multiplications.



s_1
s_2
State
3
s_N
1
2
3
$T + 1$
Time, t



s_1
$\alpha_1(t)$
s_2
$\alpha_2(t)$
s_3
$\alpha_3(t)$
s_N
$\alpha_N(t)$
s_j
$\alpha_j(t+1)$
$a_{1j}b_{1jo_t}$
$a_{2j}b_{2jo_t}$
$a_{3j}b_{3jo_t}$
$a_{Nj}b_{Njo_t}$
t
$t+1$

6.4. The Three Fundamental Questions for HMM

The backward procedure

$$\beta_i(t) = P(o_t \cdots o_T | X_t = i, \mu)$$

1. Initialization

$$\beta_i(T+1) = 1, \quad 1 \le i \le N$$

2. Induction

$$\beta_i(t) = \sum_{j=1}^{N} a_{ij} b_{ijo_t} \beta_j(t+1), \quad 1 \le t \le T, 1 \le i \le N$$

3. Total

$$P(O|\mu) = \sum_{i=1}^{N} \pi_i \beta_i(1)$$

Combining them

More generally, in fact, we can use any combination of forward and backward caching to work out the probability of an observation sequence. Observe that:

$$\begin{aligned}
P(O, X_t = i|\mu) &= P(o_1 \ldots o_T, X_t = i|\mu) \\
&= P(o_1 \cdots o_{t-1}, X_t = i, o_t \ldots o_T|\mu) \\
&= P(o_1 \cdots o_{t-1}, X_t = i|\mu) \\
&\quad \times P(o_t \ldots o_T|o_1 \ldots o_{t-1}, X_t = i, \mu) \\
&= P(o_1 \cdots o_{t-1}, X_t = i|\mu) P(o_t \ldots o_T|X_t = i, \mu) \\
&= \alpha_i(t)\beta_i(t)
\end{aligned}$$

Therefore:

$$P(O|\mu) = \sum_{i=1}^{N} \alpha_i(t)\beta_i(t), \quad 1 \le t \le T+1$$

6.4. The Three Fundamental Questions for HMM

Fundamental question 2:

Finding the best state sequence that best explains the observations.

Viterbi algorithm

Commonly we want to find the most likely complete path, that is:

$$\arg\max_{X} P(X|O,\mu)$$

To do this, it is sufficient to maximize for a fixed O:

$$\arg\max_{X} P(X,O|\mu)$$

An efficient trellis algorithm for computing this path is the *Viterbi algorithm*. Define:

$$\delta_j(t) = \max_{X_1 \cdots X_{t-1}} P(X_1 \cdots X_{t-1}, o_1 \cdots o_{t-1}, X_t = j|\mu)$$

This variable stores for each point in the trellis the probability of the most probable path that leads to that node. The corresponding variable $\psi_j(t)$ then records the node of the incoming arc that led to this most probable path. Using dynamic programming, we calculate the most probable path through the whole trellis as follows:

1. Initialization

$$\delta_j(1) = \pi_j, \quad 1 \le j \le N$$

2. Induction

$$\delta_j(t+1) = \max_{1 \le i \le N} \delta_i(t) a_{ij} b_{ijo_t}, \quad 1 \le j \le N$$

Store backtrace

$$\psi_j(t+1) = \arg\max_{1 \le i \le N} \delta_i(t) a_{ij} b_{ijo_t}, \quad 1 \le j \le N$$

3. Termination and path readout (by backtracking). The most likely state sequence is worked out from the right backwards:

$$\hat{X}_{T+1} = \arg\max_{1 \le i \le N} \delta_i(T+1)$$

$$\hat{X}_t = \psi_{\hat{X}_{t+1}}(t+1)$$

$$P(\hat{X}) = \max_{1 \le i \le N} \delta_i(T+1)$$

6.4. The Three Fundamental Questions for HMM

Fundamental question 3: Parameter estimation

Given a certain observation sequence, we want to find the values of the model parameters $\mu = (A, B, \pi)$ which best explain what we observed. Using Maximum Likelihood Estimation, that means we want to find the values that maximize $P(O|\mu)$:

$$\arg\max_{\mu} P(O_{\text{training}}|\mu)$$

There is no known analytic method to choose μ to maximize $P(0|\mu)$. But we can locally maximize it by an iterative hill-climbing algorithm. This algorithm is the *Baum-Welch* or *Forward-Backward algorithm*, which is a special case of the Expectation Maximization method which we will cover

Define $p_t(i, j), 1 \leq t \leq T, 1 \leq i, j \leq N$ as shown below. This is the probability of traversing a certain arc at time t given observation sequence O; see figure 9.7.

$$
\begin{aligned}
p_t(i, j) &= P(X_t = i, X_{t+1} = j | O, \mu) \\
&= \frac{P(X_t = i, X_{t+1} = j, O | \mu)}{P(O|\mu)} \\
&= \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{N} \alpha_m(t) \beta_m(t)} \\
&= \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{N} \sum_{n=1}^{N} \alpha_m(t) a_{mn} b_{mno_t} \beta_n(t+1)}
\end{aligned}
$$

Note that $\gamma_i(t) = \sum_{j=1}^{N} p_t(i, j)$.

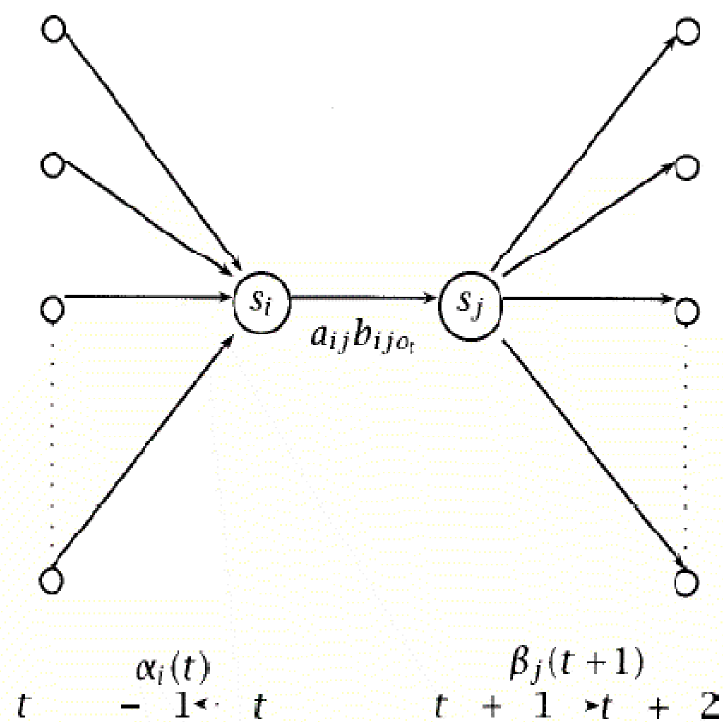


Figure 9.7 The probability of traversing an arc. Given an observation seque
and a model, we can work out the probability that the Markov process went fr
state s_i to s_j at time t.

Now, if we sum over the time index, this gives us expectations (counts):

$$\sum_{t=1}^{T} \gamma_i(t) = \text{expected number of transitions from state } i \text{ in } 0$$

$$\sum_{t=1}^{T} p_t(i,j) = \text{expected number of transitions from state } i \text{ to } j \text{ in } 0$$

$$\begin{aligned} \hat{\pi}_i &= \text{expected frequency in state } i \text{ at time } t = 1 \\ &= \gamma_i(1) \end{aligned}$$

$$\begin{aligned} \hat{a}_{ij} &= \frac{\text{expected number of transitions from state } i \text{ to j}}{\text{expected number of transitions from state } i} \\ &= \frac{\sum_{t=1}^{T} p_t(i,j)}{\sum_{t=1}^{T} \gamma_i(t)} \end{aligned}$$

$$\begin{aligned} \hat{b}_{ijk} &= \frac{\text{expected number of transitions from } i \text{ to j with } k \text{ observed}}{\text{expected number of transitions from } i \text{ to j}} \\ &= \frac{\sum_{\{t:o_t=k, 1\le t\le T\}} p_t(i,j)}{\sum_{t=1}^{T} p_t(i,j)} \end{aligned}$$

$$P(O|\hat{\mu}) \ge P(O|\mu)$$

This is a general property of the EM algorithm