CS344: Introduction to Artificial Intelligence (associated lab: CS386)

Pushpak Bhattacharyya
CSE Dept.,
IIT Bombay

Lecture 8: HMM; Viterbi

24th Jan, 2011

HMM Definition

- Set of states : S where |S|=N
- Output Alphabet : O where |O|=K
- Transition Probabilities : $A = \{a_{ij}\}$
- Emission Probabilities : $B = \{b_j(o_k)\}$
- Initial State Probabilities : π

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

Markov Processes

- Properties
  - Limited Horizon: Given previous *t* states, a state *i*, is independent of preceding *0* to *t-k+1* states.
    - $P(X_t=i|X_{t-1}, X_{t-2}, \ldots X_0) = P(X_t=i|X_{t-1}, X_{t-2} \ldots X_{t-k})$
    - Order *k* Markov process
  - Time invariance: (shown for *k=1*)
    - $P(X_t=i|X_{t-1}=j) = P(X_1=i|X_0=j) \ldots = P(X_n=i|X_{n-1}=j)$

Three basic problems (contd.)

- Problem 1: Likelihood of a sequence
  - Forward Procedure
  - Backward Procedure
- Problem 2: Best state sequence
  - Viterbi Algorithm
- Problem 3: Re-estimation
  - Baum-Welch (Forward-Backward Algorithm)

Probabilistic Inference

- O: Observation Sequence
- S: State Sequence

- Given O find S* where $S^* = \arg\max_S p(S/O)$ called Probabilistic Inference

- Infer "Hidden" from "Observed"
- How is this inference different from logical inference based on propositional or predicate calculus?

Essentials of Hidden Markov Model

1. Markov + Naive Bayes

2. Uses both transition and observation probability

$$p(S_k \xrightarrow{O_k} S_{k+1}) = p(O_k / S_k)\, p(S_{k+1} / S_k)$$

3. Effectively makes Hidden Markov Model a Finite State Machine (FSM) with probability

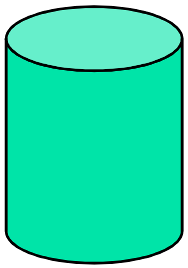
Probability of Observation Sequence

$$p(O) = \sum_{S} p(O, S)$$
$$= \sum_{S} p(S)\, p(O / S)$$

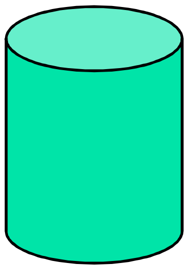
- Without any restriction,
  - Search space size= $|S|^{|O|}$

Continuing with the Urn example

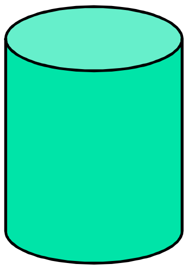
Colored Ball choosing



Urn 1
\# of Red = 30
\# of Green = 50
\# of Blue = 20



Urn 2
\# of Red = 10
\# of Green = 40
\# of Blue = 50



Urn 3
\# of Red =60
\# of Green =10
\# of Blue = 30

Example (contd.)

Given :

Transition Probability

	U_1	U_2	U_3
U_1	0.1	0.4	0.5
U_2	0.6	0.2	0.2
U_3	0.3	0.4	0.3

and

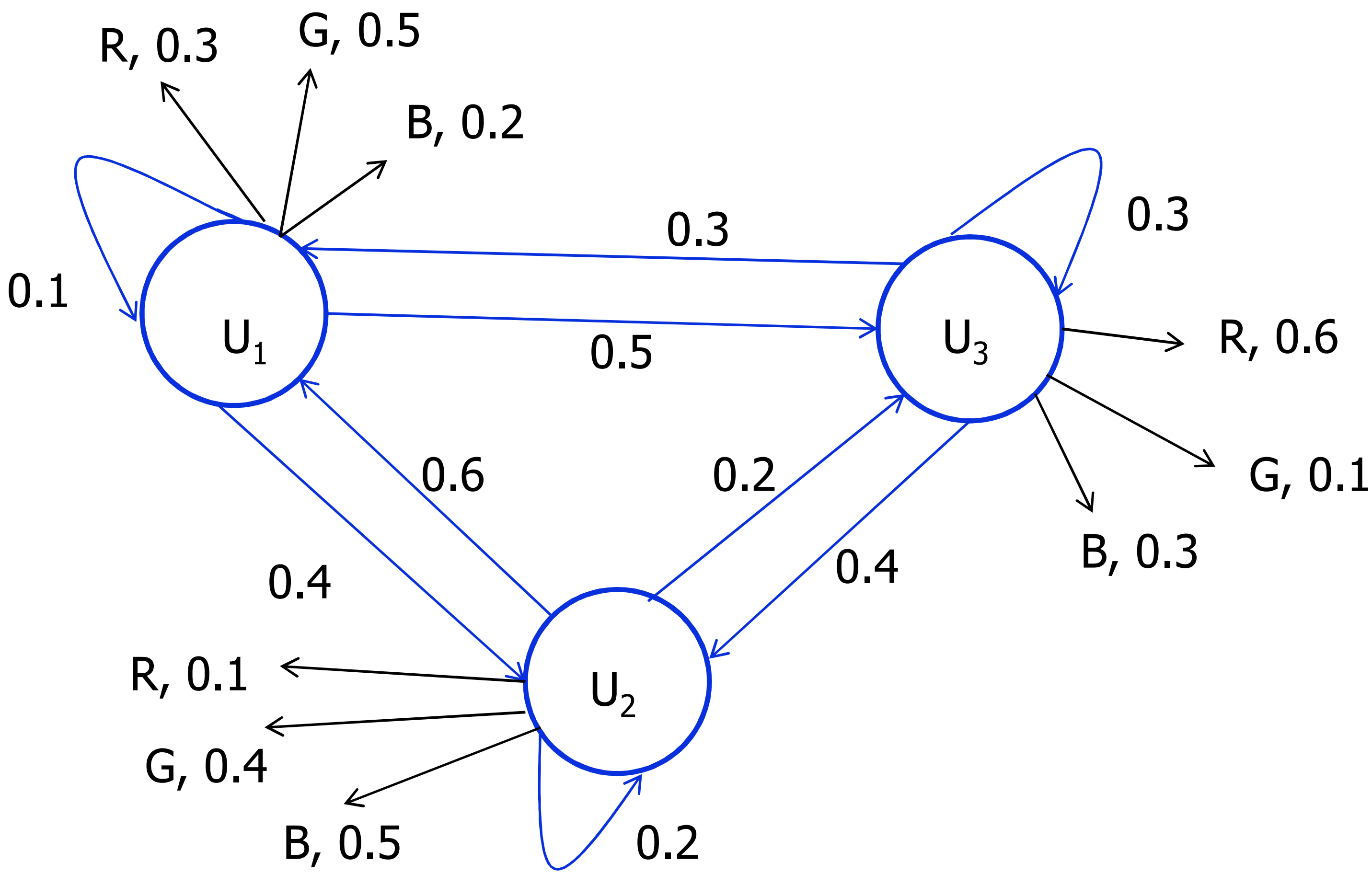
Observation/output Probability

	R	G	B
U_1	0.3	0.5	0.2
U_2	0.1	0.4	0.5
U_3	0.6	0.1	0.3

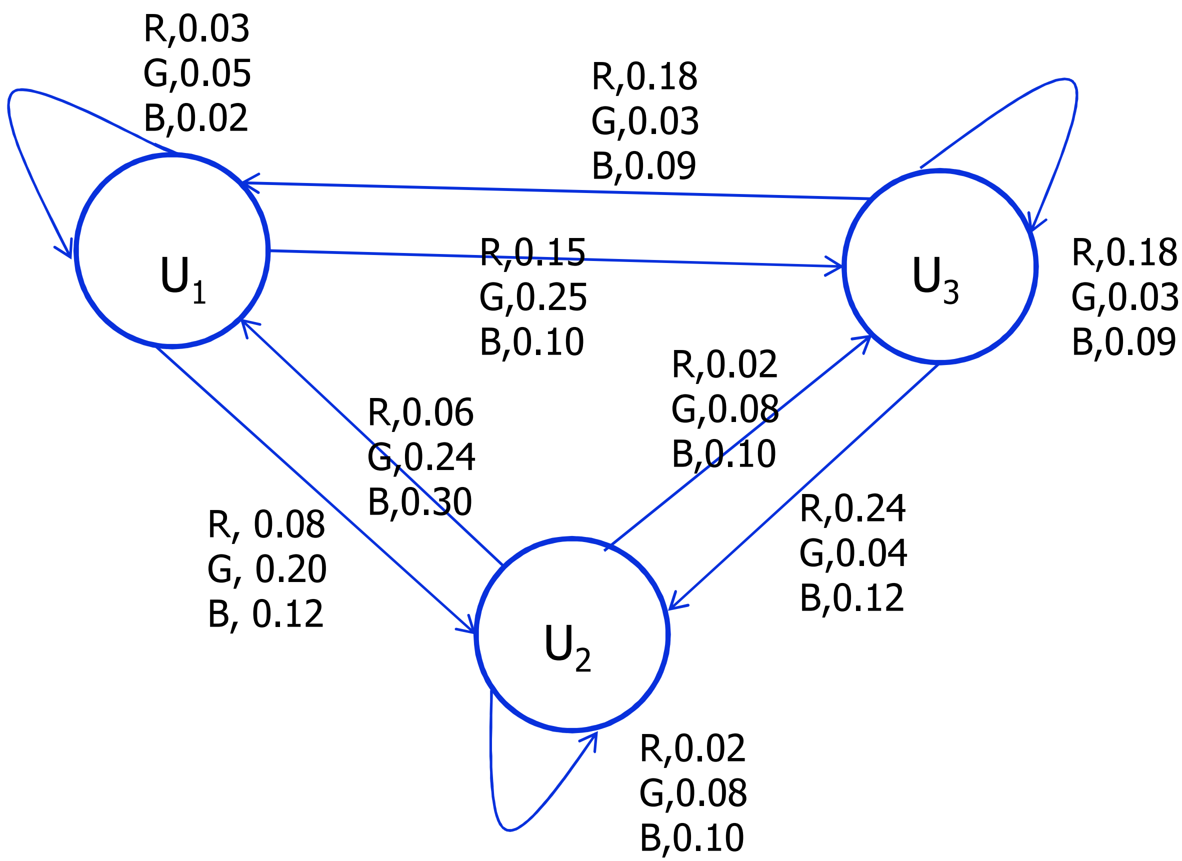
Observation : RRGGBRGR

What is the corresponding state sequence ?

Diagrammatic representation (1/2)



Diagrammatic representation (2/2)



Observations and states

	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8
OBS:	R	R	G	G	B	R	G	R
State:	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8

$S_i = U_1/U_2/U_3$; A particular state

S: State sequence

O: Observation sequence

S* = "best" possible state (urn) sequence

Goal: Maximize P(S*|O) by choosing "best" S

Goal

- Maximize P(S|O) where S is the State Sequence and O is the Observation Sequence

$$S^* = \arg\max_S (P(S \mid O))$$

Baye's Theorem

$$P(A \mid B) = P(A).P(B \mid A) / P(B)$$

P(A) -: Prior
P(B|A) -: Likelihood

$$\operatorname{argmax}_S P(S \mid O) = \operatorname{argmax}_S P(S).P(O \mid S)$$

State Transitions Probability

$$P(S) = P(S_{1-8})$$

$$P(S) = P(S_1).P(S_2 | S_1).P(S_3 | S_{1-2}).P(S_4 | S_{1-3})..P(S_8 | S_{1-7})$$

By Markov Assumption (k=1)

$$P(S) = P(S_1).P(S_2 | S_1).P(S_3 | S_2).P(S_4 | S_3)..P(S_8 | S_7)$$

Observation Sequence probability

$$P(O|S) = P(O_1|S_{1-8}).P(O_2|O_1,S_{1-8}).P(O_3|O_{1-2},S_{1-8})..P(O_8|O_{1-7},S_{1-8})$$

Assumption that ball drawn depends only on the Urn chosen

$$P(O|S) = P(O_1|S_1).P(O_2|S_2).P(O_3|S_3)...P(O_8|S_8)$$

$$P(S|O) = P(S).P(O|S)$$

$$P(S|O) = P(S_1).P(S_2|S_1).P(S_3|S_2).P(S_4|S_3)...P(S_8|S_7).$$

$$P(O_1|S_1).P(O_2|S_2).P(O_3|S_3)...P(O_8|S_8)$$

Grouping terms

	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	
Obs:	ε	R	R	G	G	B	R	G	R	
State:	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9

$$
\begin{aligned}
&P(S).P(O|S) \\
&= [P(O_0|S_0).P(S_1|S_0)]. \\
&\quad [P(O_1|S_1).\quad P(S_2|S_1)]. \\
&\quad [P(O_2|S_2).\quad P(S_3|S_2)]. \\
&\quad [P(O_3|S_3).P(S_4|S_3)]. \\
&\quad [P(O_4|S_4).P(S_5|S_4)]. \\
&\quad [P(O_5|S_5).P(S_6|S_5)]. \\
&\quad [P(O_6|S_6).P(S_7|S_6)]. \\
&\quad [P(O_7|S_7).P(S_8|S_7)]. \\
&\quad [P(O_8|S_8).P(S_9|S_8)].
\end{aligned}
$$

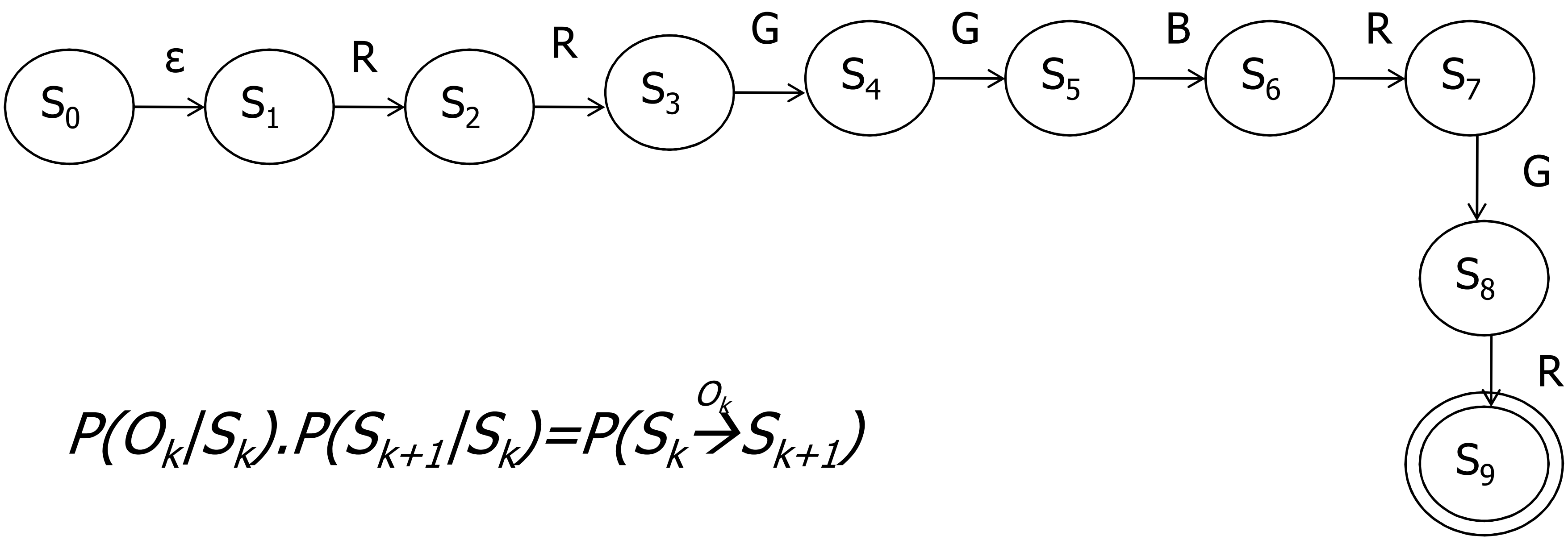
We introduce the states S_0 and S_9 as initial and final states respectively.

After S_8 the next state is S_9 with probability 1, i.e., $P(S_9|S_8)=1$

O_0 is ε-transition

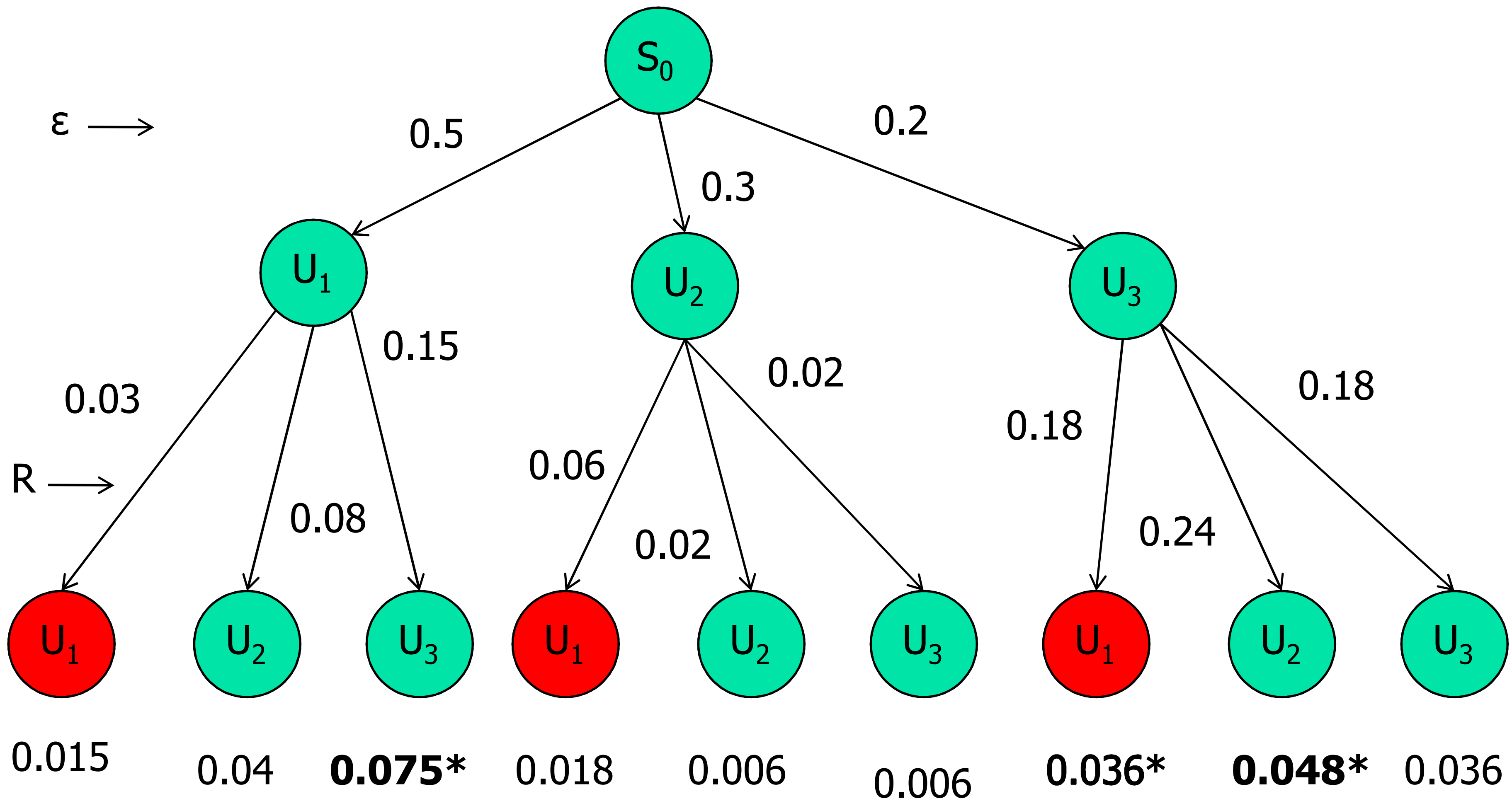
Introducing useful notation

	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	
Obs:	ε	R	R	G	G	B	R	G	R	
State:	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9



$$P(O_k|S_k).P(S_{k+1}|S_k)=P(S_k \xrightarrow{O_k} S_{k+1})$$

Viterbi Algorithm for the Urn problem (first two symbols)



***: winner sequences**

Markov process of order>1 (say 2)

	O_0	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	
Obs:	ε	R	R	G	G	B	R	G	R	
State:	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9

Same theory works

$P(S).P(O|S)$

$= P(O_0|S_0).P(S_1|S_0).$
$[P(O_1|S_1).\quad P(S_2|S_1S_0)].$
$[P(O_2|S_2).\quad P(S_3|S_2S_1)].$
$[P(O_3|S_3).P(S_4|S_3S_2)].$
$[P(O_4|S_4).P(S_5|S_4S_3)].$
$[P(O_5|S_5).P(S_6|S_5S_4)].$
$[P(O_6|S_6).P(S_7|S_6S_5)].$
$[P(O_7|S_7).P(S_8|S_7S_6)].$
$[P(O_8|S_8).P(S_9|S_8S_7)].$

We introduce the states S_0 and S_9 as initial and final states respectively.

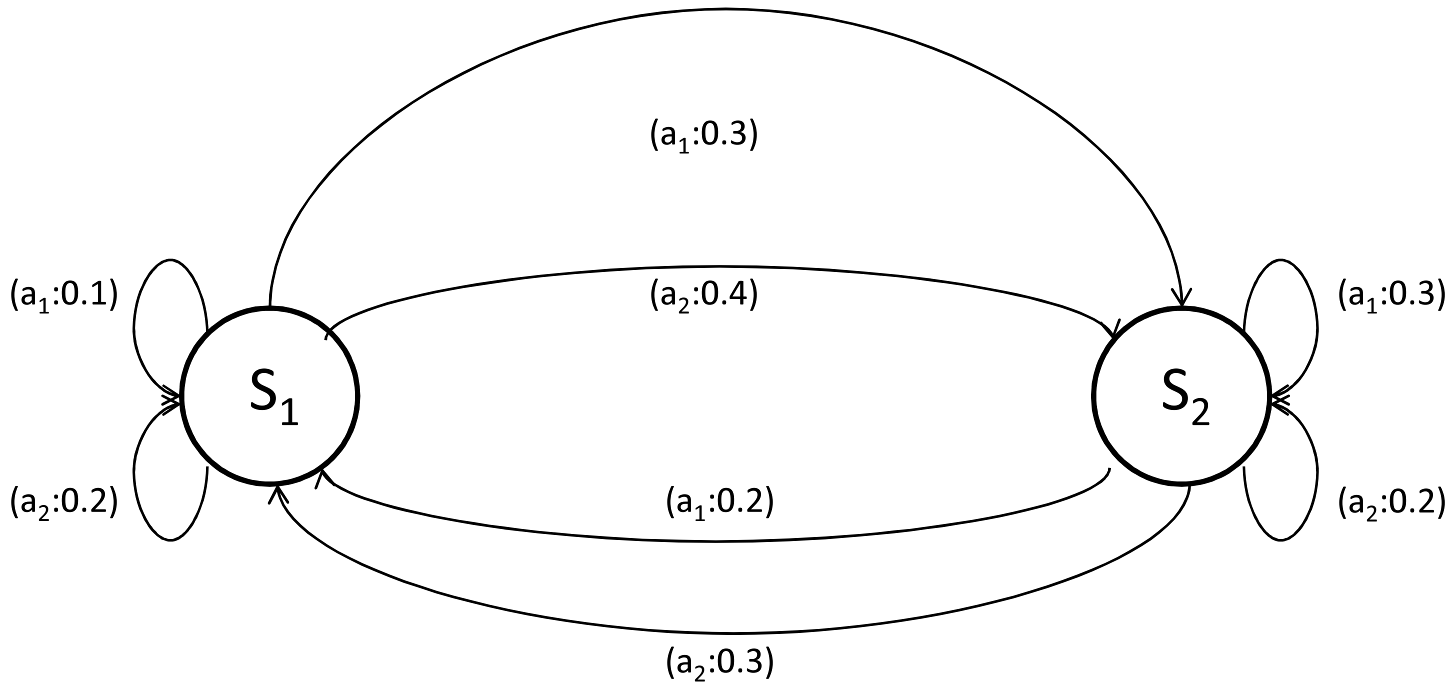
After S_8 the next state is S_9 with probability 1, i.e., $P(S_9|S_8S_7)=1$

O_0 is ε-transition

Adjustments

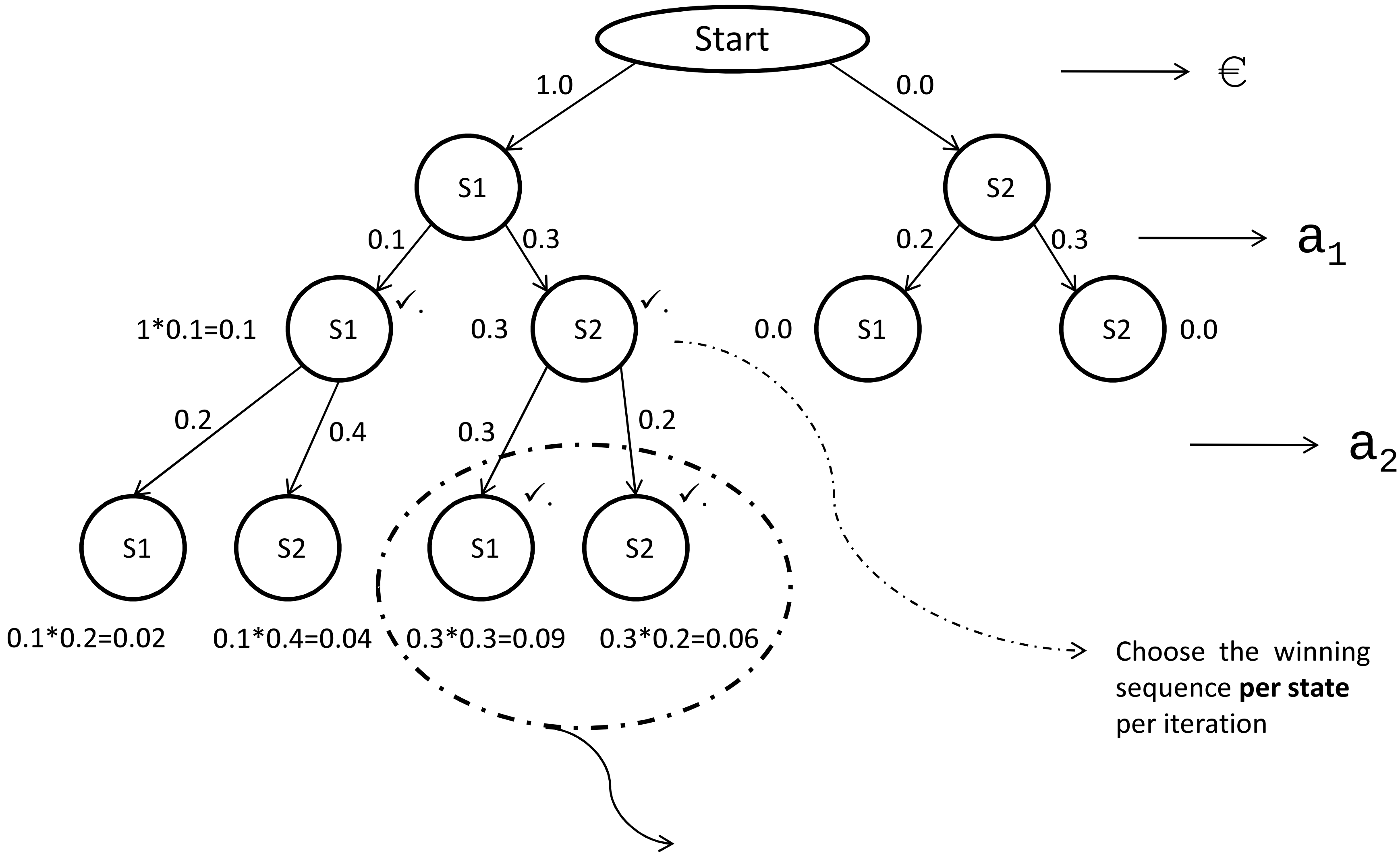
- Transition probability table will have tuples on rows and states on columns
- Output probability table will remain the same
- In the Viterbi tree, the Markov process will take effect from the 3rd input symbol (εRR)
- There will be 27 leaves, out of which **only 9 will remain**
- Sequences ending in **same tuples** will be compared
  - Instead of U1, U2 and U3
  - U_1U_1, U_1U_2, U_1U_3, U_2U_1, U_2U_2,U_2U_3, U_3U_1,U_3U_2,U_3U_3

Probabilistic FSM



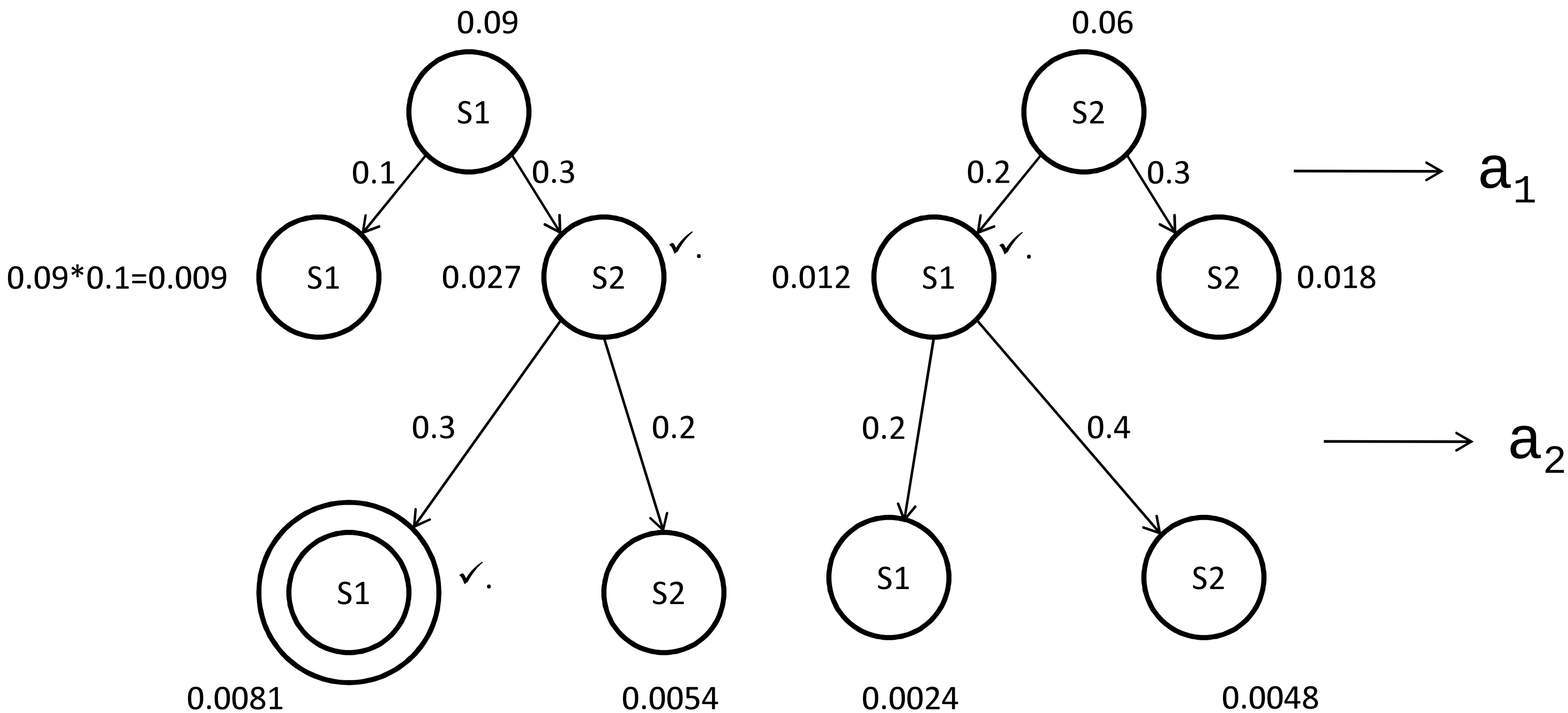
The question here is:
"what is the most likely state sequence given the output sequence seen"

Developing the tree



Choose the winning sequence **per state** per iteration

Tree structure contd…



The problem being addressed by this tree is $S^* = \arg\max_{s} P(S \mid a_1 - a_2 - a_1 - a_{2,\mu})$

a1-a2-a1-a2 is the output sequence and μ the model or the machine

Path found:
(working backward)

$$S_1 \xrightarrow{a_1} S_2 \xrightarrow{a_2} S_1 \xrightarrow{a_1} S_2 \xrightarrow{a_2} S_1$$

Problem statement: Find the best possible sequence

$$S^* = \arg\max_{S} P(S \mid O, \mu)$$

$where,\ S \rightarrow \text{State Seq},\ O \rightarrow \text{Output Seq},\ \mu \rightarrow \text{Model or Machine}$

$$\text{Model or Machine} = \{S_0, S, A, T\}$$

Start symbol　State collection　Alphabet set　Transitions

$$\text{T is defined as}\quad P(S_i \xrightarrow{a_k} S_j) \qquad \forall_{i,\ j,\ k}$$

Tabular representation of the tree

Latest symbol observed / Ending state	€	a_1	a_2	a_1	a_2
S_1	1.0	(1.0*0.1,0.0*0.2)=(**0.1**,0.0)	(0.02, **0.09**)	(0.009, **0.012**)	(0.0024, **0.0081**)
S_2	0.0	(1.0*0.3,0.0*0.3)=(**0.3**,0.0)	(0.04,**0.06**)	(**0.027**,0.018)	(0.0048,0.0054)

Note: Every cell records the winning probability ending in that state

Final winner

The bold faced values in each cell shows the sequence probability ending in that state. Going backward from final winner sequence which ends in state S2 (indicated By the 2nd tuple), we recover the sequence.

Algorithm

(following James Alan, Natural Language Understanding (2nd edition), Benjamin Cummins (pub.), 1995

Given:

1. The HMM, which means:
   a. Start State: S_1
   b. Alphabet: $A = \{a_1, a_2, \ldots a_p\}$
   c. Set of States: $S = \{S_1, S_2, \ldots S_n\}$
   d. Transition probability $P(S_i \xrightarrow{a_k} S_j) \quad \forall_{i,j,k}$
      which is equal to $P(S_j, a_k \mid S_i)$
2. The output string $a_1 a_2 \ldots a_T$

To find:

The most likely sequence of states $C_1 C_2 \ldots C_T$ which produces the given output sequence, *i.e.*, $C_1 C_2 \ldots C_T = \arg\max_C [P(C \mid a_1, a_2, \ldots a_T, \mu)]$

Algorithm contd…

Data Structure:

1. A N*T array called SEQSCORE to maintain the winner sequence always (N=#states, T=length of o/p sequence)
2. Another N*T array called BACKPTR to recover the path.

Three distinct steps in the Viterbi implementation

1. Initialization
2. Iteration
3. Sequence Identification

1. Initialization

SEQSCORE(1,1)=1.0

BACKPTR(1,1)=0

For(i=2 to N) do

    SEQSCORE(i,1)=0.0

[expressing the fact that first state is S_1]

2. Iteration

For(t=2 to T) do

  For(i=1 to N) do

    SEQSCORE(i,t) = $\text{Max}_{(j=1,N)}$

$$[SEQSCORE(j,(t-1)) * P(Sj \xrightarrow{a_k} Si)]$$

    BACKPTR(I,t) = index *j* that gives the MAX above

3. Seq. Identification

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

For i from (T-1) to 1 do

C(i) = BACKPTR[C(i+1),(i+1)]

Optimizations possible:

1. BACKPTR can be 1*T
2. SEQSCORE can be T*2

Homework:- Compare this with A*, Beam Search [Homework]

Reason for this comparison:

Both of them work for finding and recovering sequence