HIDDEN MARKOV MODELS

UNIT – 2

Prepared By, Param Ahir

TOPICS

- Markov Chain
- Hidden Markov Chains
- Likelihood Computation
- HMM Training

MARKOV CHAINS

- Markovian Assumption states that the past doesn't give a piece of valuable information. Given the present, history is irrelevant to know what will happen in the future.
- Markov Chain is a stochastic process that follows the Markovian Assumption.

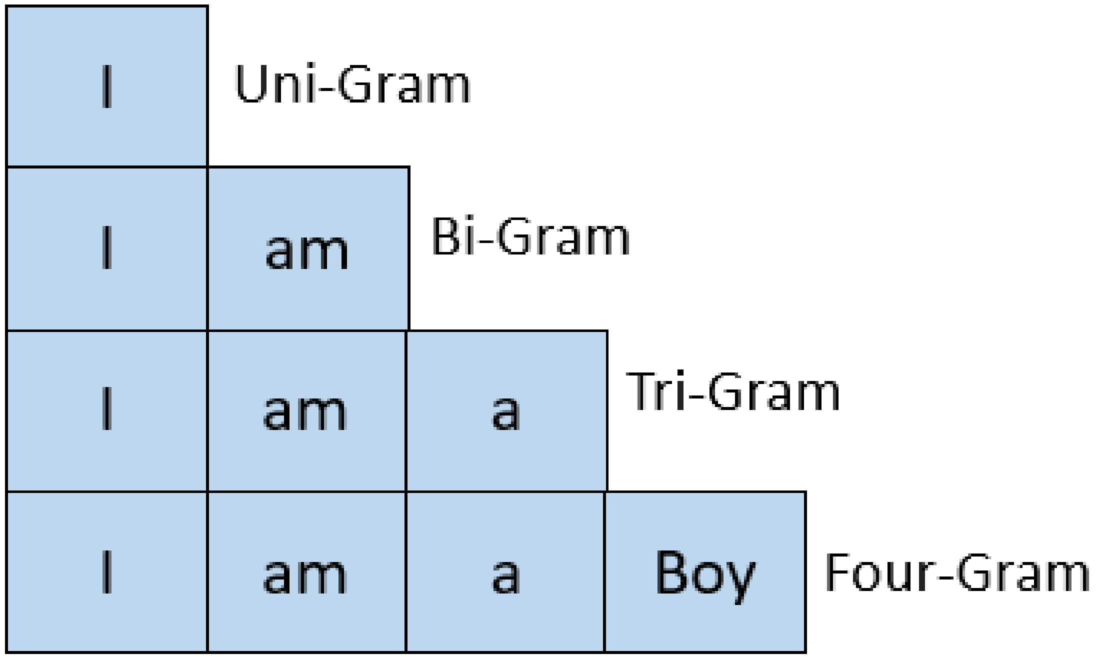
MARKOV CHAINS

MARKOV CHAINS

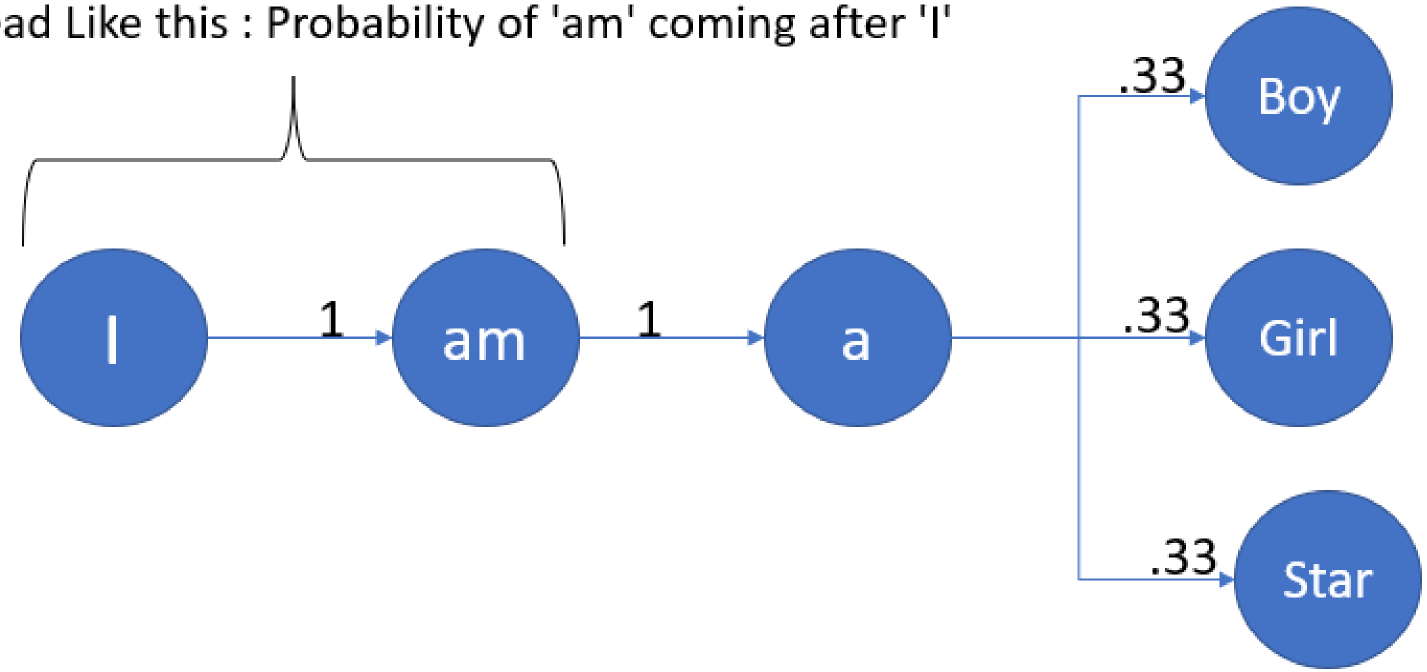
- Sentence 1: I am a Boy.
- Sentence 2: I am a Girl.
- Sentence 3: I am a Star.

MARKOV CHAINS

- Sentence 1: I am a Boy.
- Sentence 2: I am a Girl.
- Sentence 3: I am a Star.



N-Gram Concept



Current State

	I	am	a	Boy	Girl	Star
I	0	1	0	0	0	0
am	0	0	1	0	0	0
a	0	0	0	1	1	1
Boy	0	0	0	0	0	0
Girl	0	0	0	0	0	0
Star	0	0	0	0	0	0

Next State

Read Like this : After 'I' which word(s) has appeared?
Only one word that is 'am' so the probability will be 1

Read Like this : After 'a' which word(s) has appeared?
3 words appeared 'Boy','Girl','Star'.
So out of these 3 word what is the probability of those 3 words appearing after a? Ans 1/2

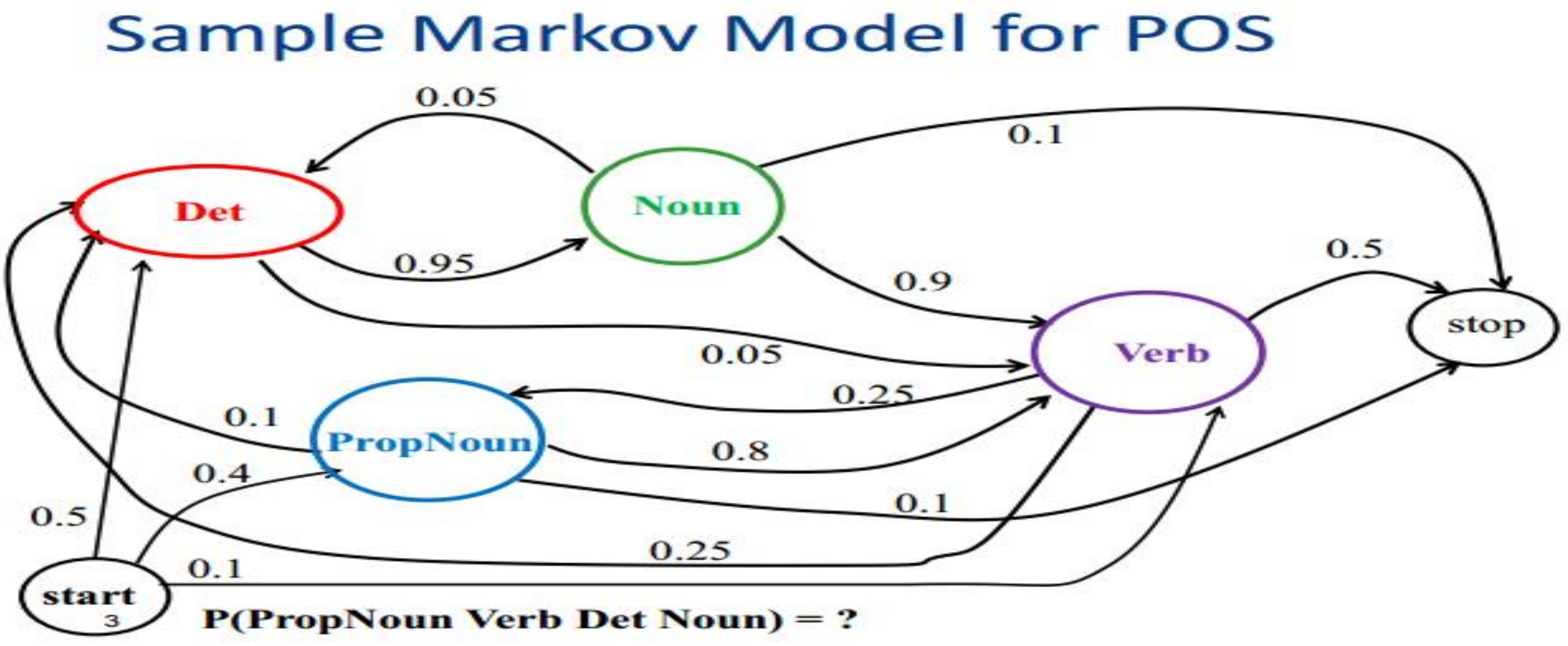
MARKOV MODEL

Markov Model (= Markov Chain)

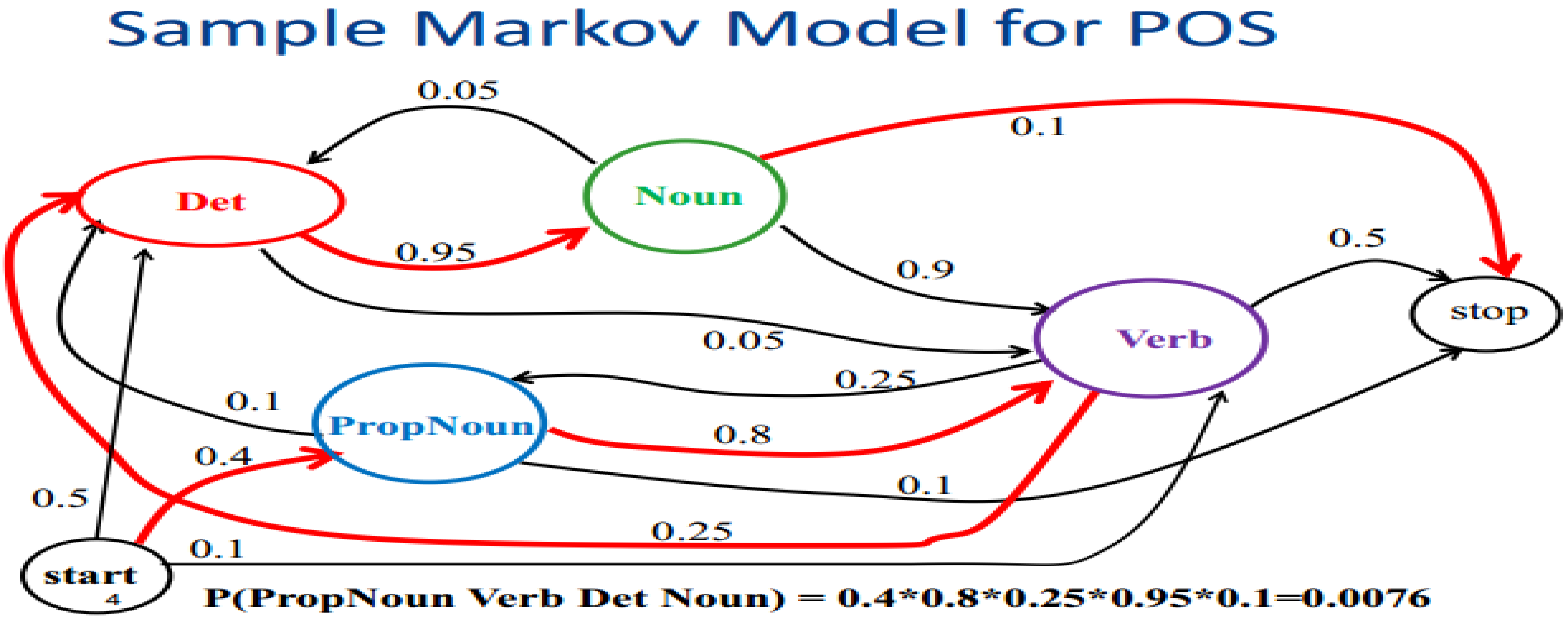
- A sequence of random variables visiting a set of states
- Transition probability specifies the probability of transiting from one state to the other.
- Language Model!
- Markov Assumption: next state depends only on the current state and independent of previous history.

$$\Pr(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \Pr(X_{n+1} = x | X_n = x_n).$$

MARKOV MODEL



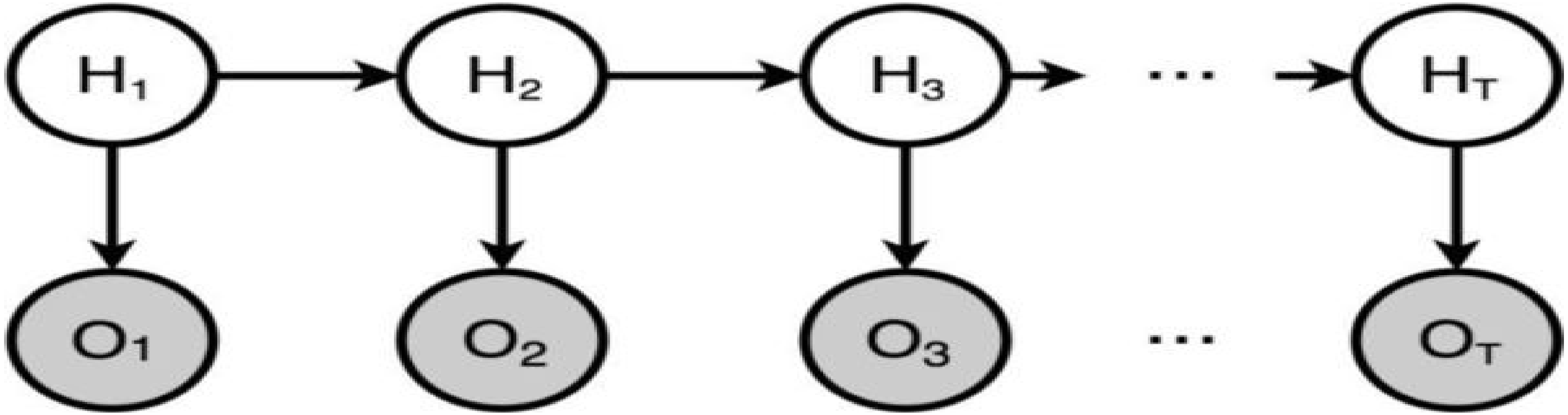
MARKOV MODEL



HIDDEN MARKOV MODEL

- Probabilistic *generative* model for sequences.
- HMM Definition with respect to POS tagging:
  - States = POS tags
  - Observation = a sequence of words
  - Transition probability = bigram model for POS tags
  - Observation probability = probability of generating each token (word) from a given POS tag
- "Hidden" means the exact sequence of states (a sequence of POS tags) that generated the observation (a sequence of words) are hidden.

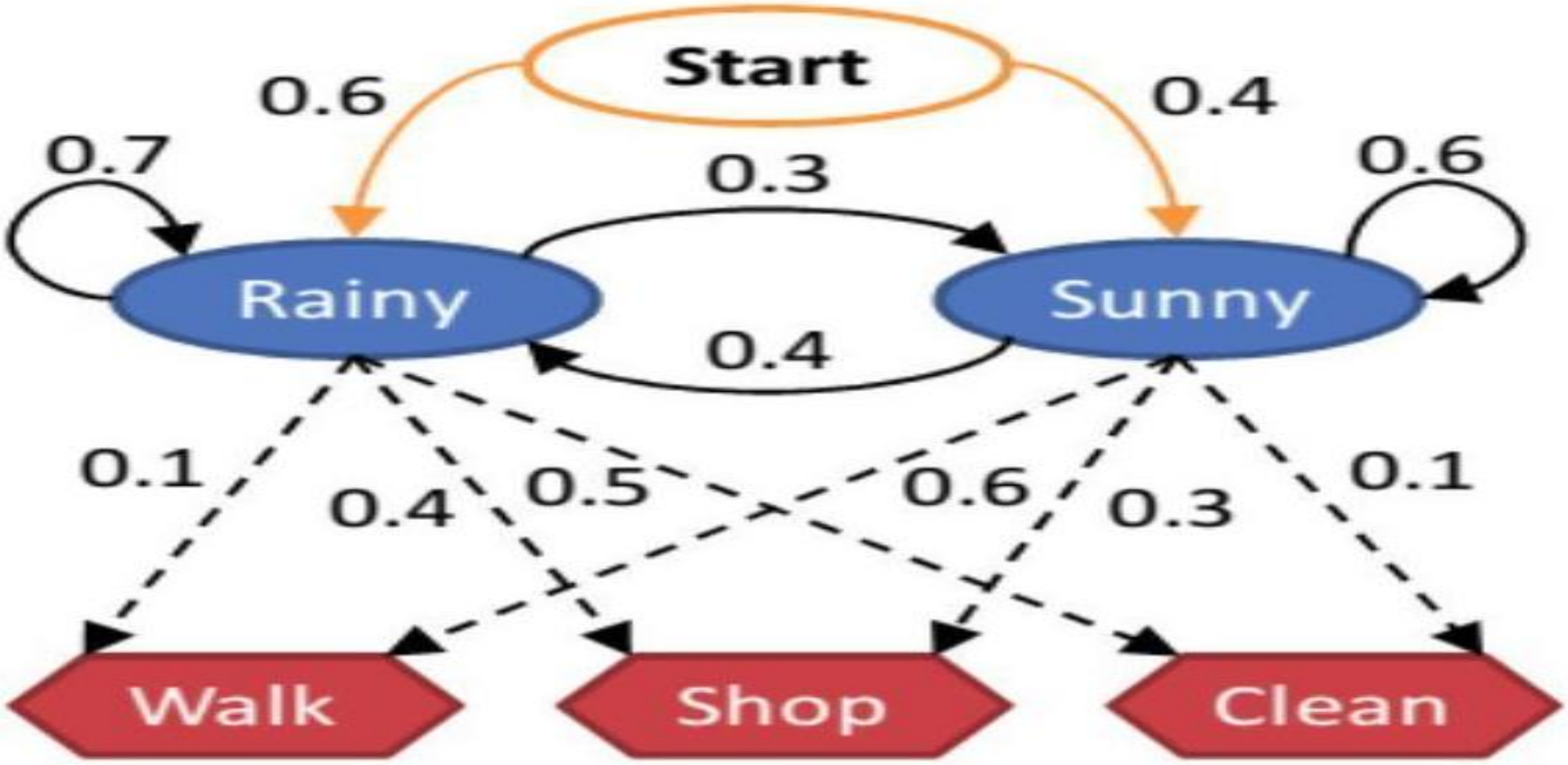
HIDDEN MARKOV MODEL



Parameters of the model are $\theta = (\pi, \mathbf{A}, \mathbf{B})$, with:

- the initial state vector π of elements $p(h_1)$
- the state transition matrix A of probabilities $p(h_t \mid h_{t-1})$
- the emission or observation matrix B of elements $p(o_t \mid h_t)$

HIDDEN MARKOV MODEL



PARAMETERS OF HIDDEN MARKOV MODEL

$Q = q_1 q_2 \dots q_N$	a set of N **states**
$A = a_{11} a_{12} \dots a_{n1} \dots a_{nn}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \dots o_T$	a sequence of T **observations**, each one drawn from a vocabulary $V = v_1, v_2, \dots, v_V$
$B = b_i(o_t)$	a sequence of **observation likelihoods**, also called **emission probabilities**, each expressing the probability of an observation o_t being generated from a state i
q_0, q_F	a special **start state** and **end (final) state** that are not associated with observations, together with transition probabilities $a_{01} a_{02} \dots a_{0n}$ out of the start state and $a_{1F} a_{2F} \dots a_{nF}$ into the end state

PARAMETERS OF HIDDEN MARKOV MODEL

Three important problems in HMM

Problem 1 (Likelihood):	Given an HMM $\lambda = (A,B)$ and an observation sequence O, determine the likelihood $P(O\|\lambda)$.
Problem 2 (Decoding):	Given an observation sequence O and an HMM $\lambda = (A,B)$, discover the best hidden state sequence Q.
Problem 3 (Learning):	Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

PARAMETERS OF HIDDEN MARKOV MODEL

Three important problems in HMM

Problem 1 (Likelihood):	Given an HMM $\lambda = (A, B)$ and an observation sequence O, determine the likelihood $P(O\|\lambda)$.
Problem 2 (Decoding):	Given an observation sequence O and an HMM $\lambda = (A, B)$, discover the best hidden state sequence Q.
Problem 3 (Learning):	Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

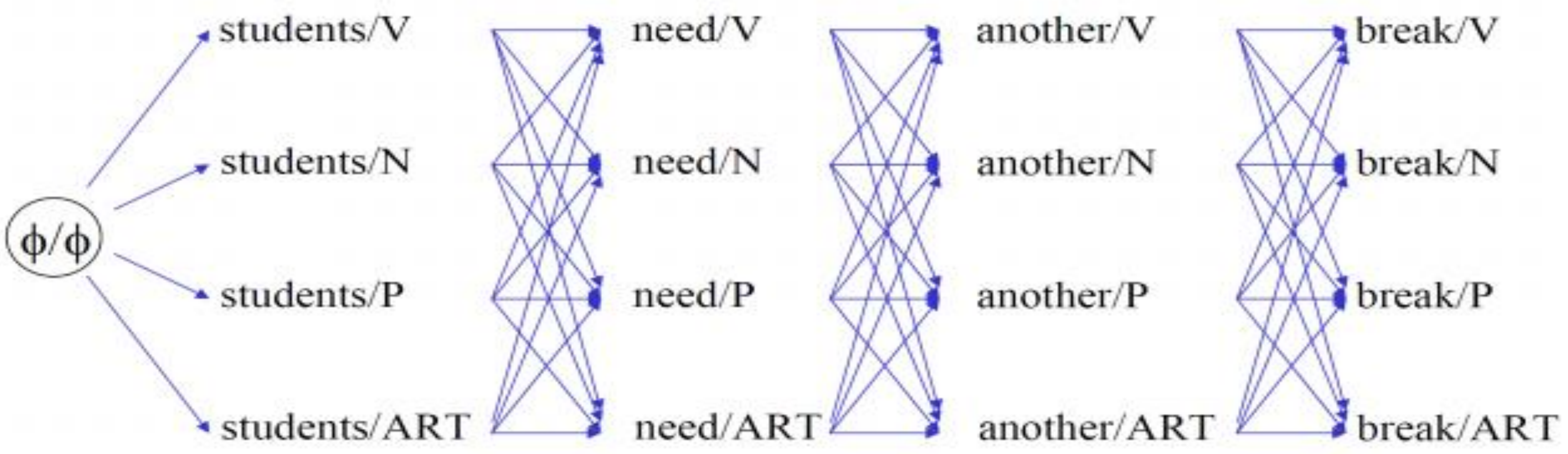
- Problem 1 (Likelihood) ➔ **Forward** Algorithm
- Problem 2 (Decoding) ➔ **Viterbi** Algorithm
- Problem 3 (Learning) ➔ **Forward-backward** Algorithm

HMM DECODING - VITERBI ALGORITHM

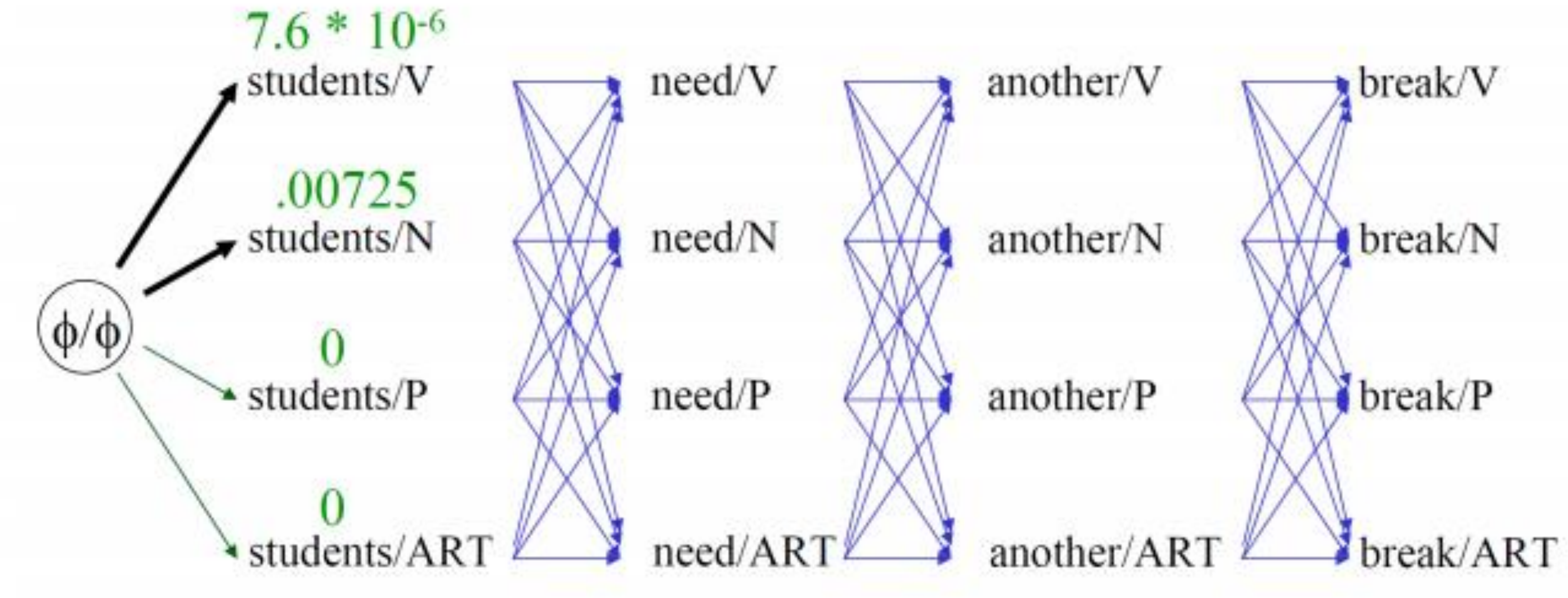
- Decoding finds the most likely sequence of states that produced the observed sequence.
  - A sequence of states = pos-tags
  - A sequence of observation = words
- Naïve solution: brute force search by enumerating all possible sequences of states.

  → problem?
- Dynamic Programming!
- Standard procedure is called the Viterbi algorithm (Viterbi, 1967) and has $O(N^2T)$ time complexity.

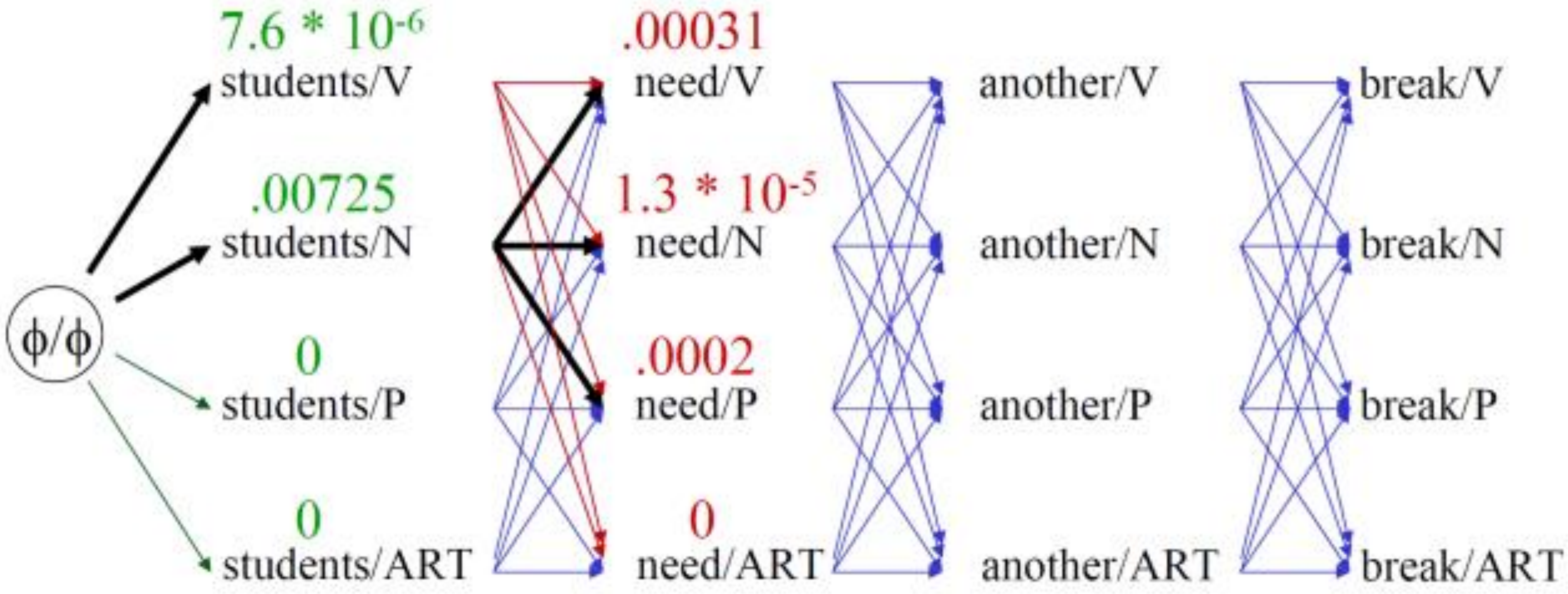
HMM DECODING - VITERBI ALGORITHM



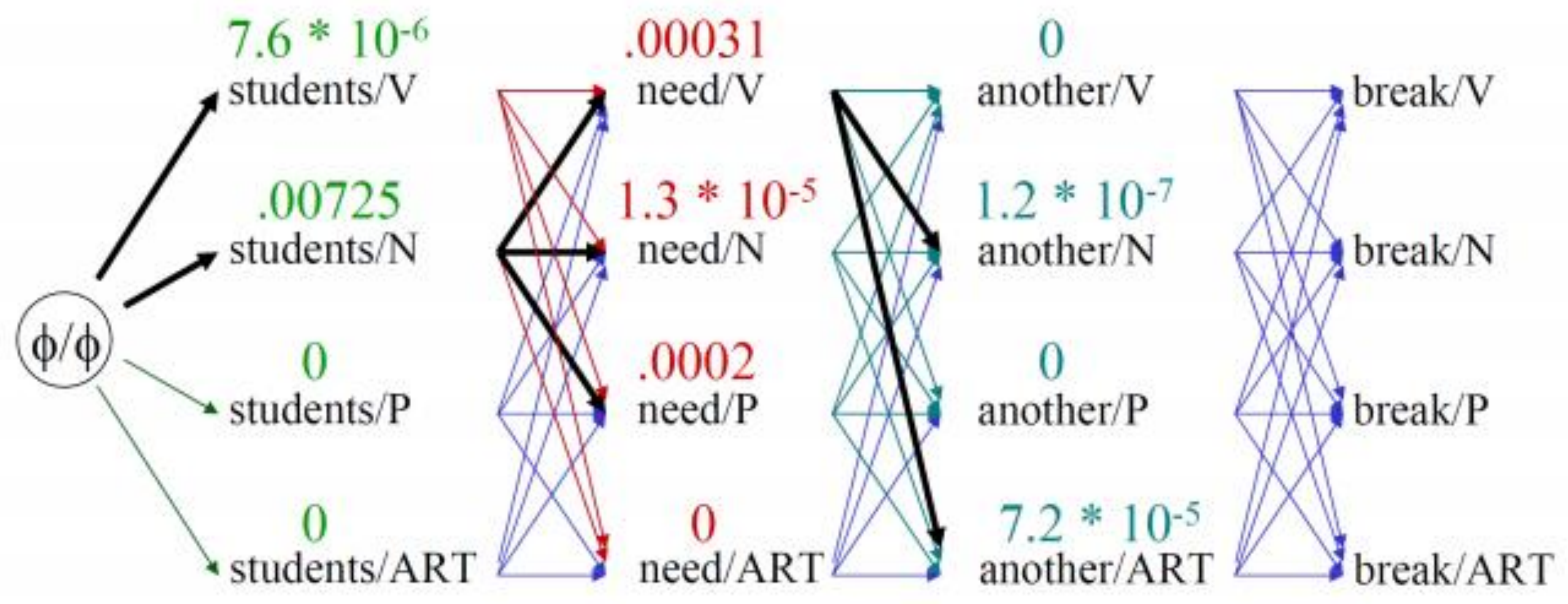
HMM DECODING - VITERBI ALGORITHM



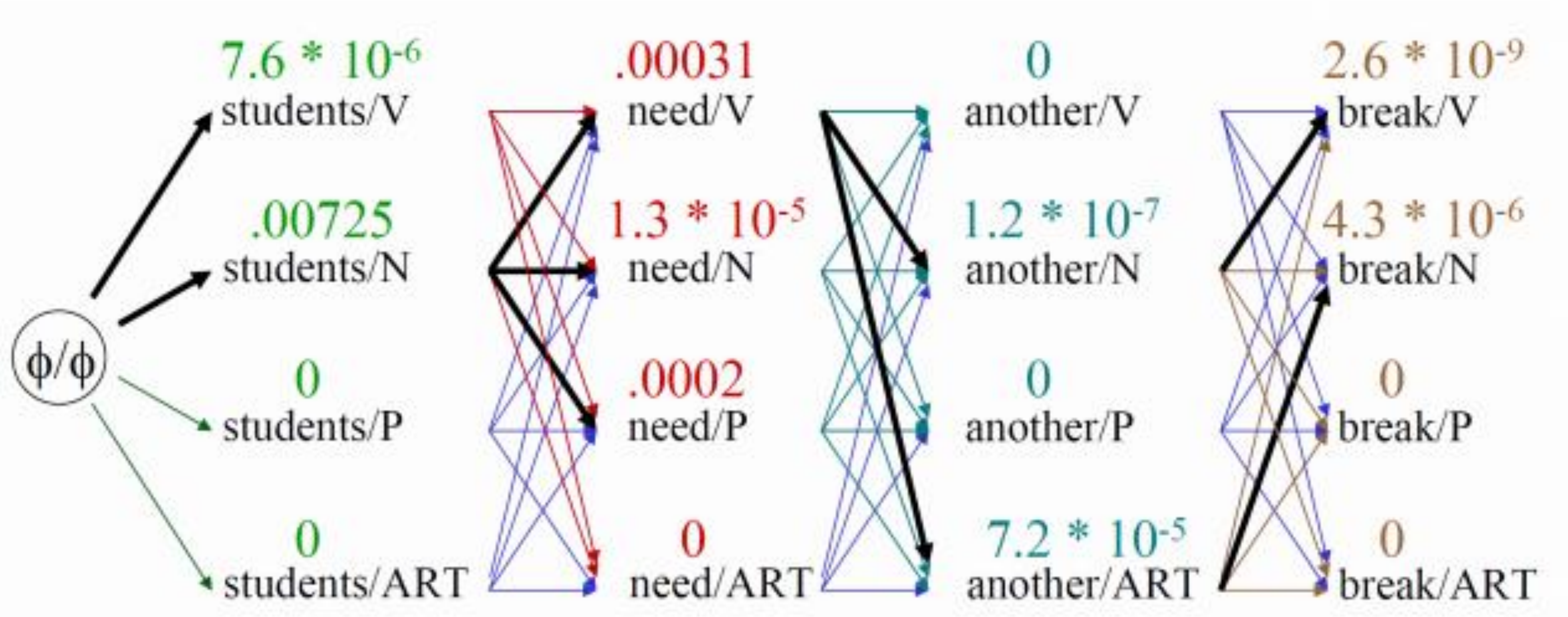
HMM DECODING - VITERBI ALGORITHM



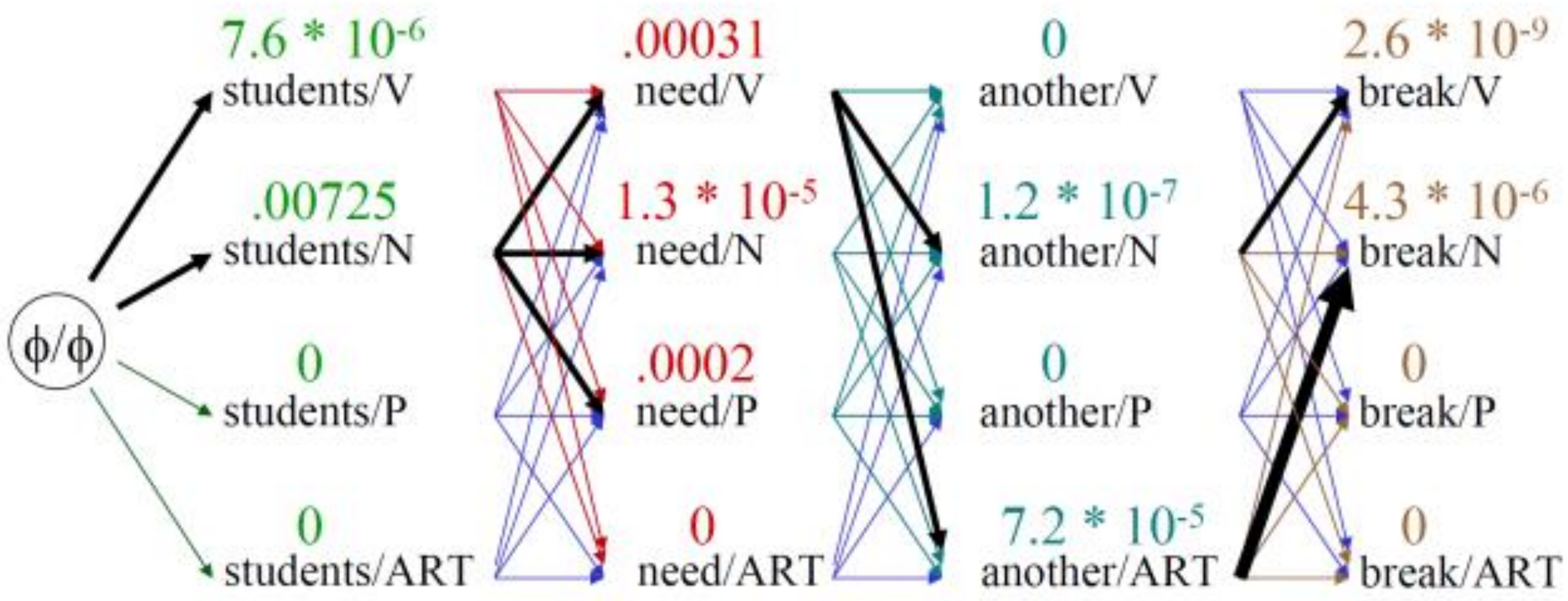
HMM DECODING - VITERBI ALGORITHM



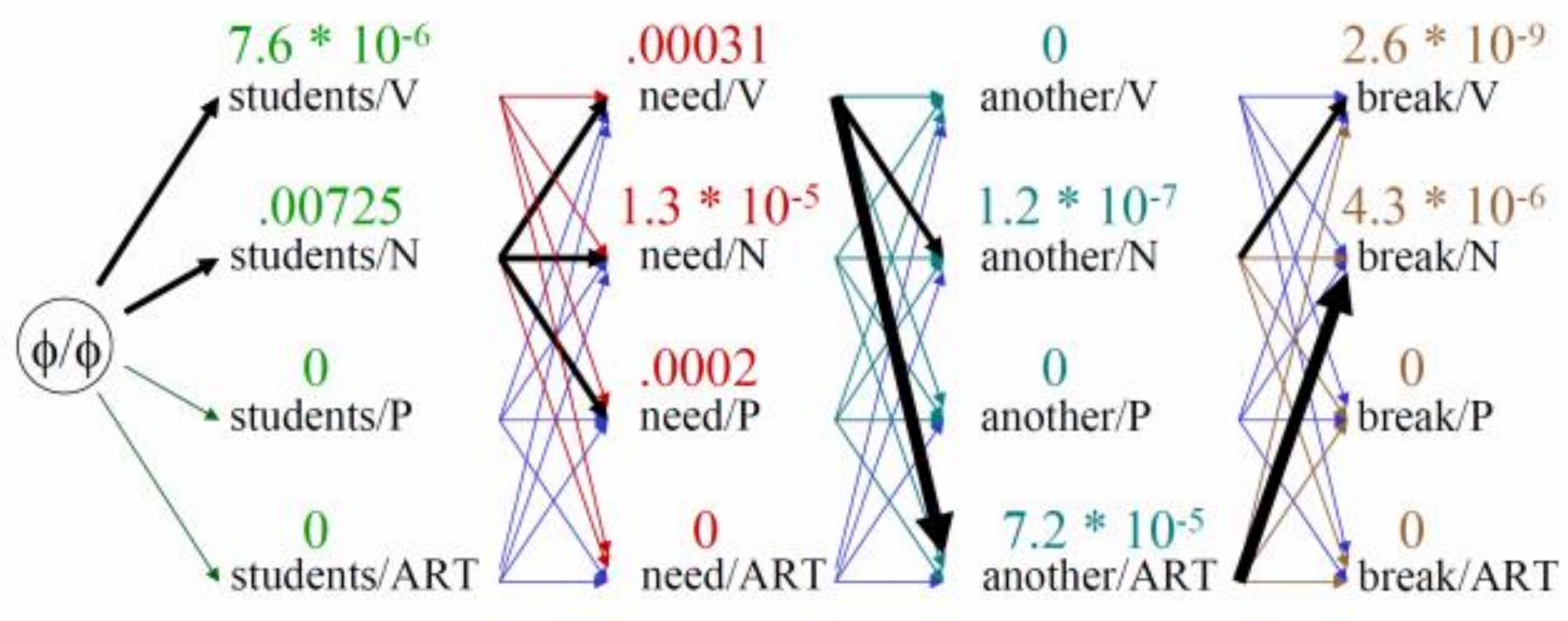
HMM DECODING - VITERBI ALGORITHM



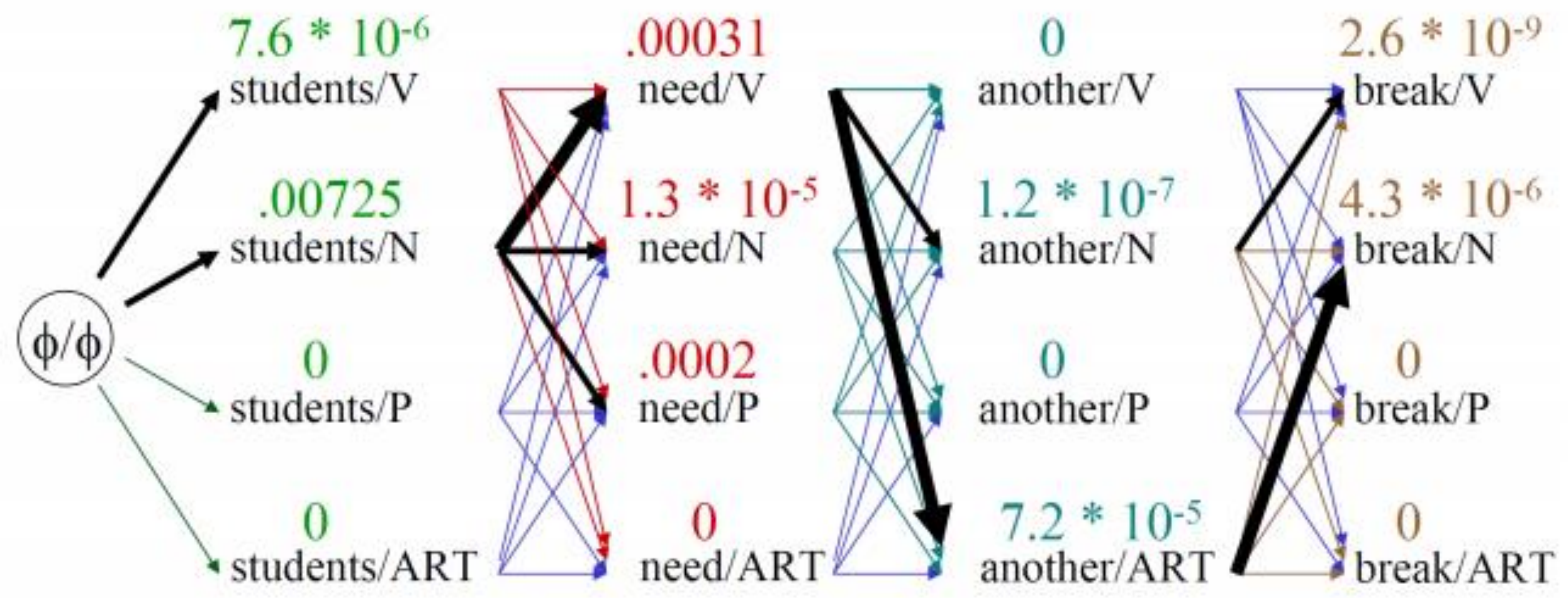
HMM DECODING - VITERBI ALGORITHM



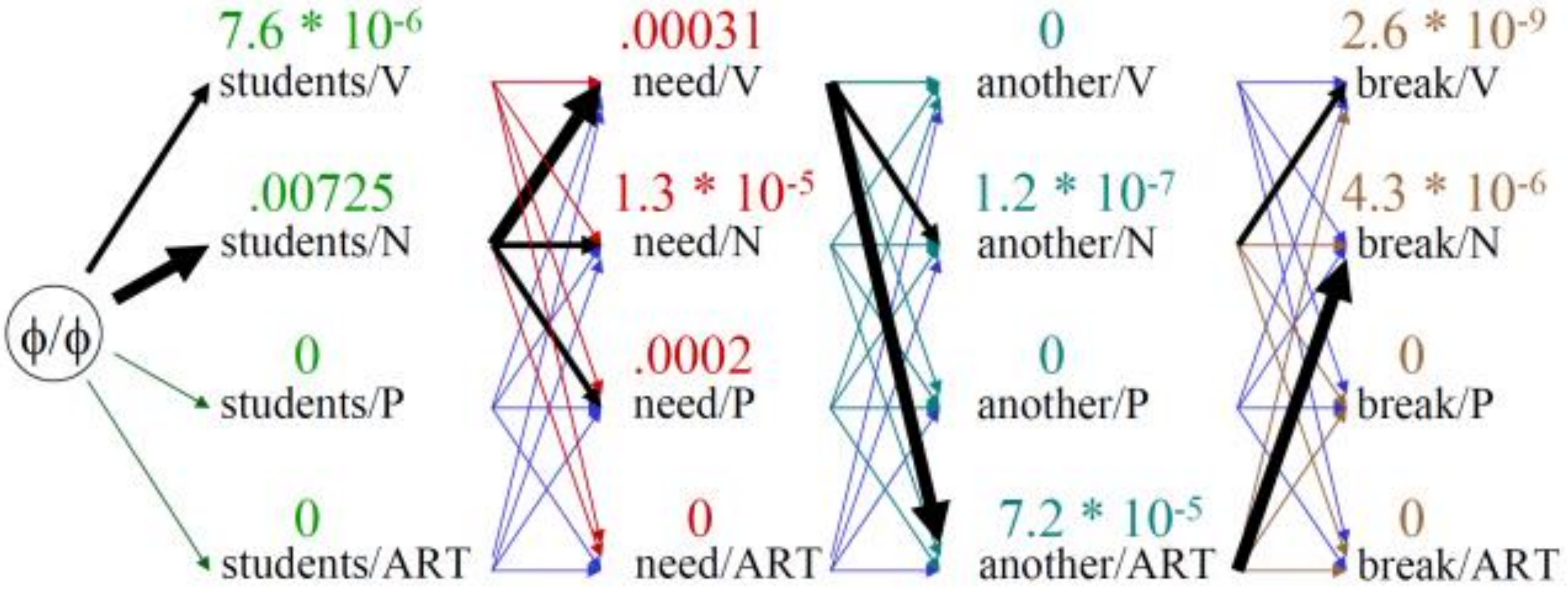
HMM DECODING - VITERBI ALGORITHM



HMM DECODING - VITERBI ALGORITHM



HMM DECODING - VITERBI ALGORITHM



HMM LIKELIHOOD - FORWARD ALGORITHM

- Given a sequence of observations, *O*, and a model with a set of parameters, λ, what is the probability that this observation was generated by this model: $P(O|\ \lambda)$?

HMM LIKELIHOOD - FORWARD ALGORITHM

- Due to the Markov assumption, the probability of being in any state at any given time t only relies on the probability of being in each of the possible states at time $t-1$.
- Forward Algorithm: Uses dynamic programming to exploit this fact to efficiently compute observation likelihood in $O(TN^2)$ time.
  - Compute a ***forward trellis*** that compactly and implicitly encodes information about all possible state paths.

HMM LIKELIHOOD - FORWARD ALGORITHM

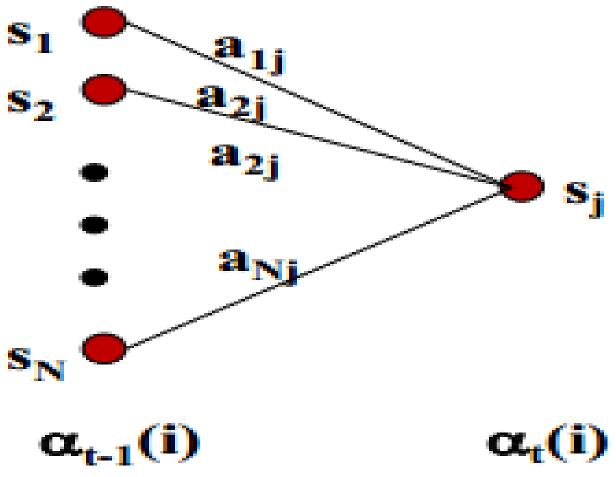
Forward Probabilities

- Let $\alpha_t(j)$ be the probability of being in state j after seeing the first t observations (by summing over all initial paths leading to j).

$$\alpha_t(j) = P(o_1, o_2, \ldots o_t,\ q_t = s_j \mid \lambda)$$

HMM LIKELIHOOD - FORWARD ALGORITHM

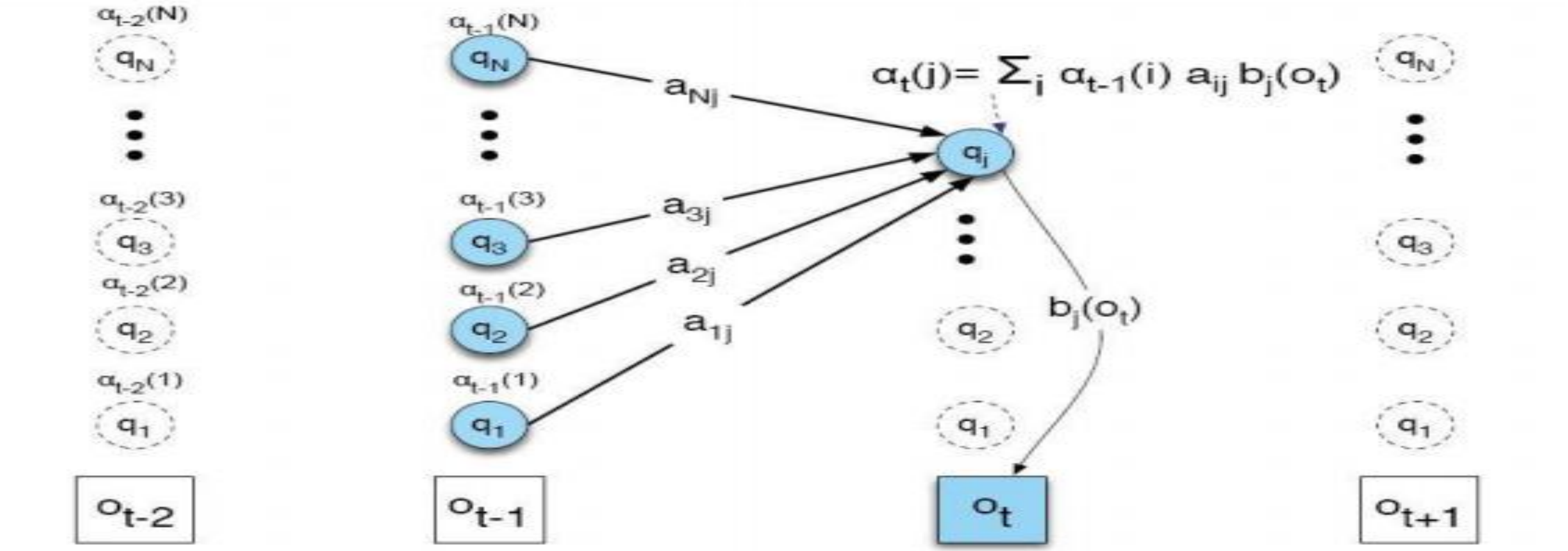
Forward Step



- Consider all possible ways of getting to s_j at time t by coming from all possible states s_i and determine probability of each.
- Sum these to get the total probability of being in state s_j at time t while accounting for the first $t-1$ observations.
- Then multiply by the probability of actually observing o_t in s_j.

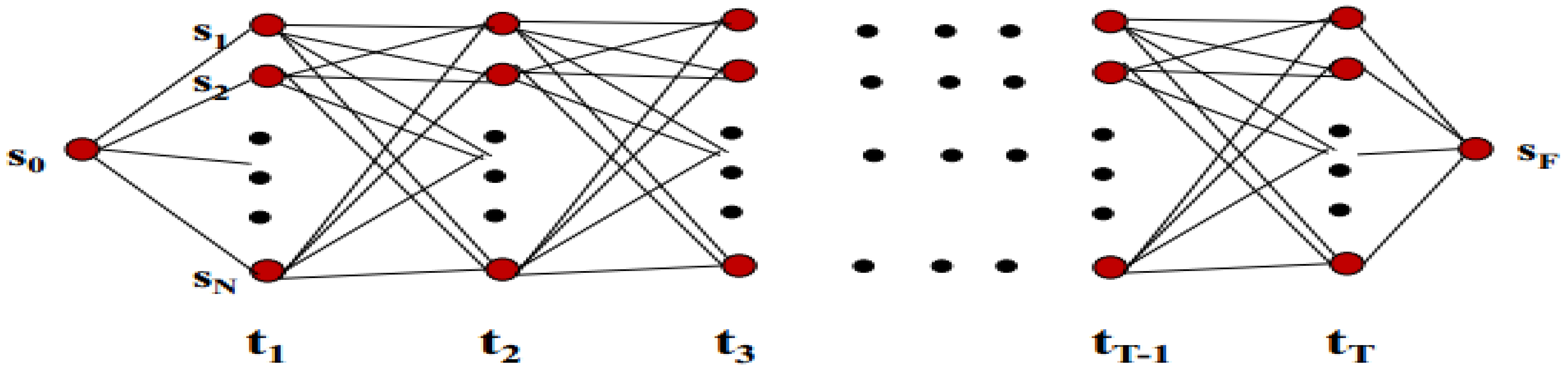
HMM LIKELIHOOD - FORWARD ALGORITHM

$\alpha_{t-1}(i)$	the **previous forward path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j



HMM LIKELIHOOD - FORWARD ALGORITHM

Forward Trellis



- Continue forward in time until reaching final time point and sum probability of ending in final state.

Forward Computational Complexity

- Requires only $O(TN^2)$ time to compute the probability of an observed sequence given a model.
- Exploits the fact that all state sequences must merge into one of the N possible states at any point in time and the Markov assumption that only the last state effects the next one.

HMM LIKELIHOOD - FORWARD ALGORITHM

Supervised Learning:

- ☐ All training sequences are completely labeled (tagged).
- ☐ That is, nothing is really "hidden" strictly speaking.
- ☐ Learning is very simple → by MLE estimate

Unsupervised Learning:

- ☐ All training sequences are unlabeled (tags are unknown)
- ☐ We do assume the number of tags, i.e. states
- ☐ True HMM case.
- ☐ Forward-Backward Algorithm

HMM LIKELIHOOD - FORWARD - BACKWARD ALGORITHM

1. Start with initial probability estimates [A,B]. Initially set equal probabilities or define them randomly.
2. Compute expectation of how often each transition/emission has been used. We will estimate latent variables [ξ,γ] (This is common approach for EM Algorithm)
3. Re-estimate the probabilities [A,B] based on those estimates (latent variable).
4. Repeat until convergence

HMM LIKELIHOOD - FORWARD - BACKWARD ALGORITHM

1) Assume an HMM with *N* states.
2) Randomly set its parameters λ=(*A*,*B*) (making sure they represent legal distributions)
3) Until converge (i.e. λ no longer changes) do:

   E Step: Use the forward/backward procedure to determine the probability of various possible state sequences for generating the training data

   M Step: Use these probability estimates to re-estimate values for all of the parameters λ

APPLICATIONS

Detection of DNA regions

- Observation: DNA sequence
- Hidden state: gene, transcription factor, protein-coding region…
- Learning: EM
- Validation often against known regions, and then through biological experiment

APPLICATIONS

Music composition

- Observations: notes played
- States: chords
- Learning: music by one composer, labelled with correct chords, used for maximum likelihood learning
- Model ”composes” by sampling chords and notes from the model
- If successful, new music is generated “in the style” of the composer

APPLICATIONS

Speech recognition

- Observations: sound wave readings
- States: phonemes
- Learning: use labelled data to initialize the model, then EM with a much larger set of speakers to further adapt the parameters
- Transcription system: use inference to determine the most likely state sequence, which provides the transcription of the word
- HMMs are the state-of-art speech recognition technology
- Can be coupled with classification, if desired, to improve recognition performance

APPLICATIONS

Classification of time series

- Use one HMM for each class, and learn its parameters from data
- When given a new observation sequence, compute its likelihood under each HMM
- The example is assigned the label of the class that yields the highest likelihood