COMP9318: HIDDEN MARKOV MODEL

Wei Wang, University of New South Wales

Outline

- Markov Model
- Hidden Markov Model
  - Definition and basic problems
  - Decoding
  - Proj1

Applications

- On-line handwriting recognition
- Speech recognition
- Gesture recognition
- Language modeling
- Motion video analysis and tracking
- Protein sequence/gene sequence alignment
- Stock price prediction
- …

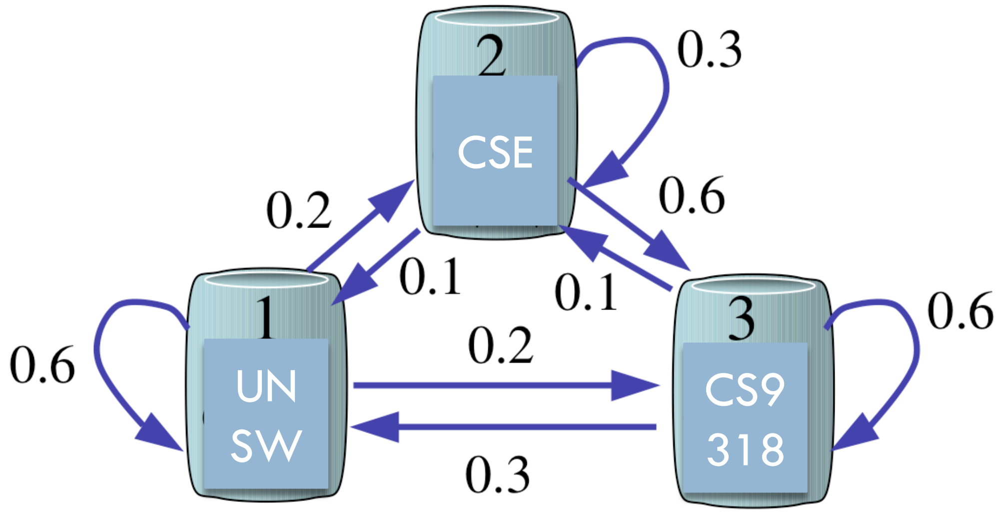
What's HMM?

- Hidden Markov Model
  - Hidden
  - Markov

Markov Model

- The model (and some notations):
  - States: Q: $\{q_0, q_1, \ldots q_{N-1}\}$
  - State sequence: $X = \{x_i\}$; each x_i takes a value in Q
  - State transition probabilities: $A_{u \rightarrow v}$
  - Initial state distribution: π
- Markov assumption (order = 1)
  - $\Pr[x_{i+1} | x_0, x_1, \ldots, x_i] = \Pr[x_{i+1} | x_i]$
  - Limited memory

Example



- Google's PageRank:
  - States: webpages
  - State sequence: sequence of webpages one visited
  - State transition probabilities:
    - $A_{u \to v}$ = #-outlinks-from-page-u-to-v / #-out-links-at-page-u
    - (actually a bit more complex)
  - Initial state distribution: π = uniform on all pages
- Markov assumption (order = 1)
  - $\Pr[x_{i+1} \mid x_0, x_1, \ldots, x_i] = \Pr[x_{i+1} \mid x_i]$
  - Randomly click an out link on page i

Another example is the n-gram Language Model (See Naïve Bayes classifier for text)

Sequence Probability

- What's the probability of the state sequence being $Q = q_0\ q_1\ \ldots\ q_T$?
- Chain rule + Markov assumption
  - $\Pr[Q|\text{model}=\lambda] = \Pr[q_0|\lambda]$

$$* \Pr[q_1|q_0,\lambda]$$

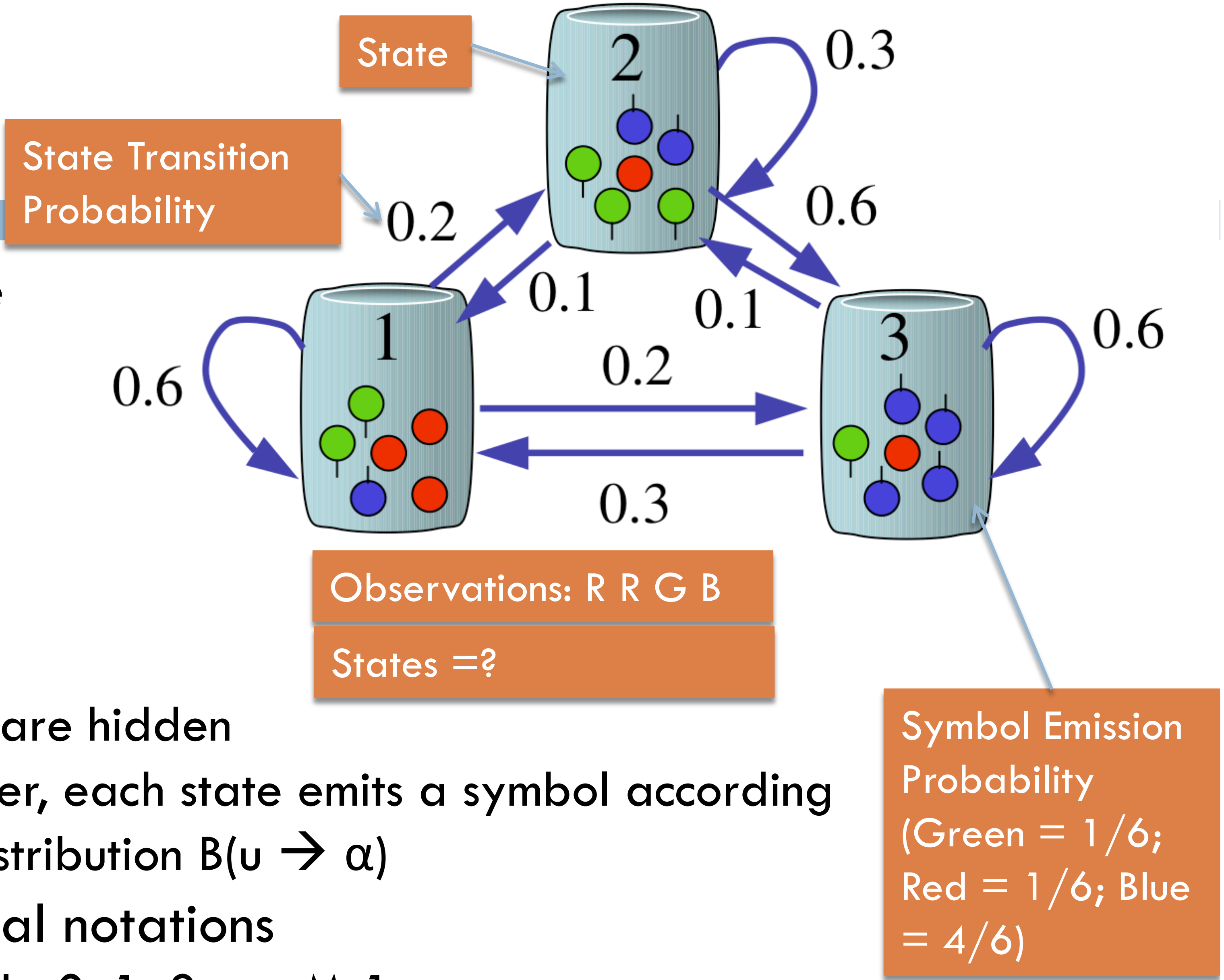
$$* \Pr[q_2|q_0,q_1,\lambda] * \ldots$$

$$* \Pr[q_T|q_0,q_1,\ldots,q_{T-1},\lambda]$$

$$= \Pr[q_0|\lambda] * (\Pr[q_1|q_0,\lambda]*\Pr[q_2|q_1,\lambda]*\ldots*\Pr[q_T|q_{T-1},\lambda])$$

HMM

- Example



- Hidden:
  - States are hidden
  - However, each state emits a symbol according to a distribution B(u → α)
- Additional notations
  - Symbols: 0, 1, 2, …, M-1
  - Observed symbol sequence: $O_0, O_1, \ldots, O_{T-1}$

The Generative Process

- Loop
  - Pick the next state the transit to
  - Transit to the chosen state, and generate an output symbol
- All according to the pmf of the distributions

$$
\begin{array}{ccccccc}
S1 & \longrightarrow & S2 & \longrightarrow & S1 & \longrightarrow & S3 \\
\downarrow & & \downarrow & & \downarrow & & \downarrow \\
O1 & & O2 & & O3 & & O4
\end{array}
$$

3 Problems

$$\begin{array}{ccccccc} S1 & \rightarrow & S2 & \rightarrow & S1 & \rightarrow & S3 \\ \downarrow & & \downarrow & & \downarrow & & \downarrow \\ O1 & & O2 & & O3 & & O4 \end{array}$$

- P1: Model Evaluation Problem
  - What's the probability of seeing this observation sequence, given the HMM model λ?
  - Compute $Pr[O_0, O_1, \ldots, O_{T-1} | \lambda]$

Forward algorithm

- P2: Decoding Problem

Viterbi algorithm

proj1

  - What is the most likely state sequence (Q) corresponding to this observation sequence, given the HMM model λ?
  - $\text{Argmax}_Q \; Pr[O_0, O_1, \ldots, O_{T-1} | Q=q_0, q_1, \ldots, q_{T-1}, \lambda]$
- P3: Learning the model

Baum-Welch algorithm

  - What is the most likely parameters that generates this observation sequence?

Exercise: States: 1 2 3 1 → Probability = 1/32400

Generation Probability

c.f., P8 sequence prob.

All conditioned on λ

- $Pr[O_0, O_1, \ldots, O_{T-1} | q_0, q_1, \ldots, q_{T-1}, \lambda]$

$= Pr[q_0 | \lambda] * Pr[O_0 | q_0] *$

$Pr[q_1 | q_0, \lambda] * Pr[O_1 | q_1]$

$* \ldots *$

$Pr[q_{T-1} | q_{T-2}, \lambda] * Pr[O_{T-1} | q_{T-1}]$

$= \pi[q_0] * B[q_0 \rightarrow O_0] *$

$A[q_0 \rightarrow q_1] * B[q_1 \rightarrow O_1]$

$* \ldots *$

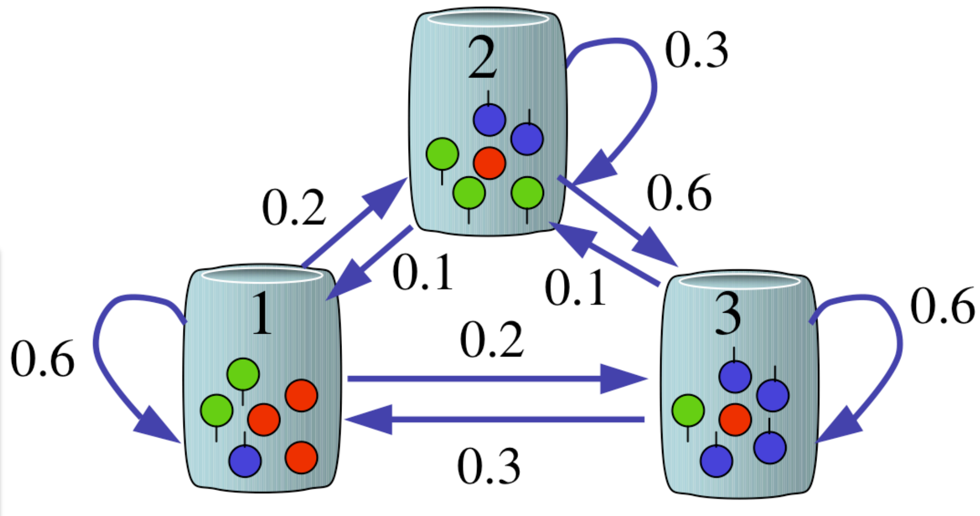
$A[q_{T-2} \rightarrow q_{T-1}] * B[q_{T-1} \rightarrow O_{T-1}]$

- Example ($\pi[q_i] = 1/3$)

States: 1 1 2 3

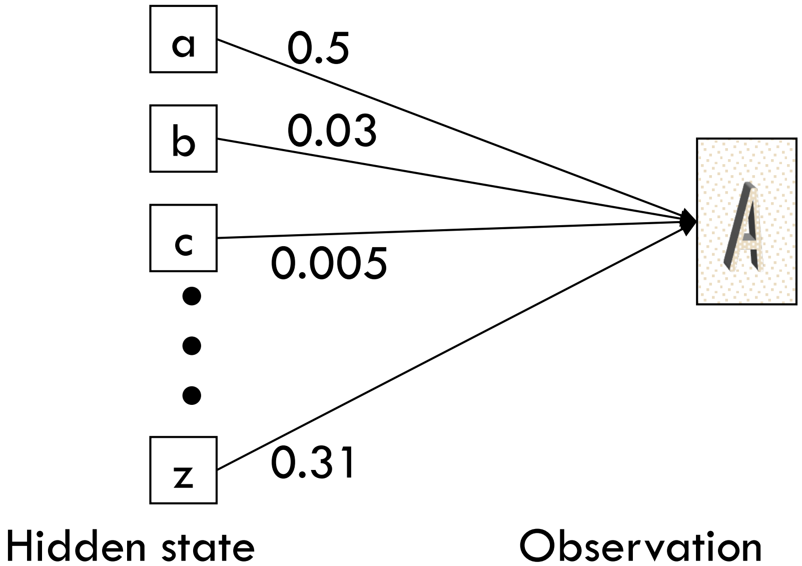
Observations: R R G B

$= (1/3)(3/6) * (0.6)(3/6) * (0.2)(3/6) * (0.6)(4/6) = 1/500$



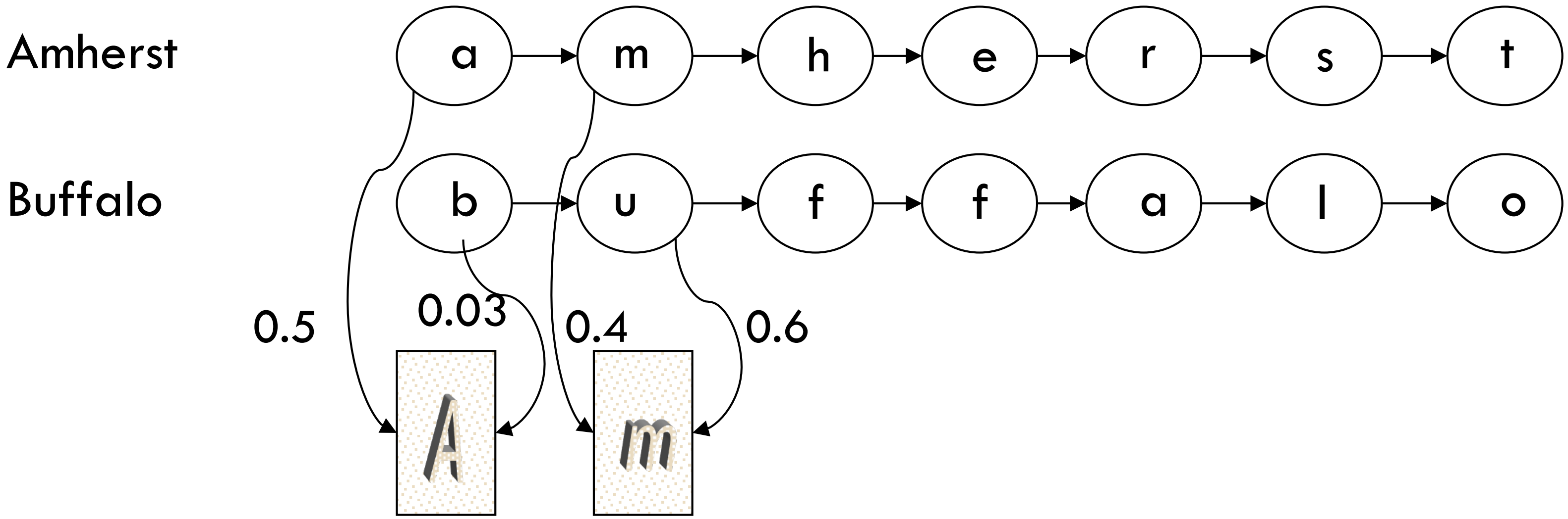
Application: Typed word recognition

- Assume all chars are separated
- Character recognizer outputs probability of the image being particular character, P(image|char).
- There are infinite number of observations though



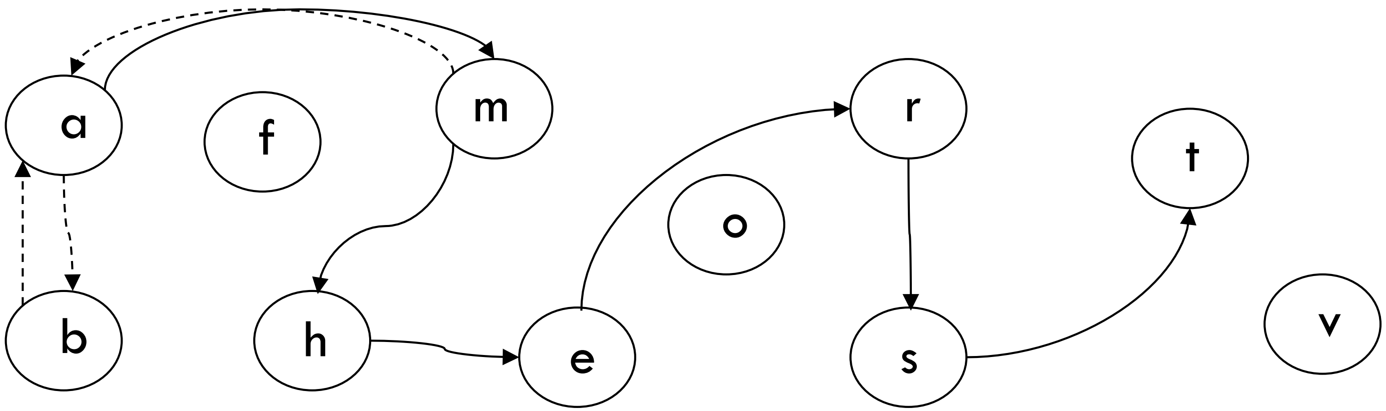
Casting into the Evaluation Problem

- Assume the lexicon is given
- Construct separate HMM models for each lexicon word
- Pick the model whose generation probability is the maximum



The Other Approach

- Construct a single HMM models for all lexicon words
- Pick the best state sequence (= char sequence) whose generation probability is the maximum
- This is actually the **decoding problem**



Decoding Problem

- Naïve algorithm:
  - Enumerate all possible state sequence and evaluate their probability of generating the observations
  - Pick the one whose resulting probability is the highest
  - Problem: time complexity = $O(N^T * T)$
- Viterbi: Dynamic programming-based method
  - Attempt: if we "magically" know best state sequence for RRG, can we know what's the best state sequence for RRGB?
    - No. (Give an counter example)
  - Remedy: best state sequence for RRGB must come from the best state sequence ending at ***some*** state for the last observation. We don't know which, but we can compute all.

Viterbi Algorithm

- Define $\delta[O_t \rightarrow q_i]$ as the best probability of any state sequence such that the symbol at timestamp t, denoted as O_t, corresponds to state q_i
- Recursive formula:

$$\delta[O_t \rightarrow q_i] = \max_{u \in [0, N-1]} (\delta[O_{t-1} \rightarrow q_u] * A[q_u \rightarrow q_i] * B[q_i \rightarrow O_t])$$

- Boundary condition:

$$\delta[O_0 \rightarrow q_i] = \pi[q_i] * B[q_i \rightarrow O_0]$$

Viterbi Algorithm

- Define $\delta[O_t \rightarrow q_i]$ as the best probability of any state sequence such that the symbol O_t corresponds to state q_i
- Recursive formula:

$$\delta[O_t \rightarrow q_i] = \max_{u \in [0,\, N-1]} \left(\delta[O_{t-1} \rightarrow q_u] * A[q_u \rightarrow q_i] * B[q_i \rightarrow O_t] \right)$$

- Boundary condition:

$$\delta[O_0 \rightarrow q_i] = \pi[q_i] * B[q_i \rightarrow O_0]$$

- Easy to find the computing order in DP is from O_0 to O_{T-1}, within which we loop over all the states.

Example of Viterbi Algorithm

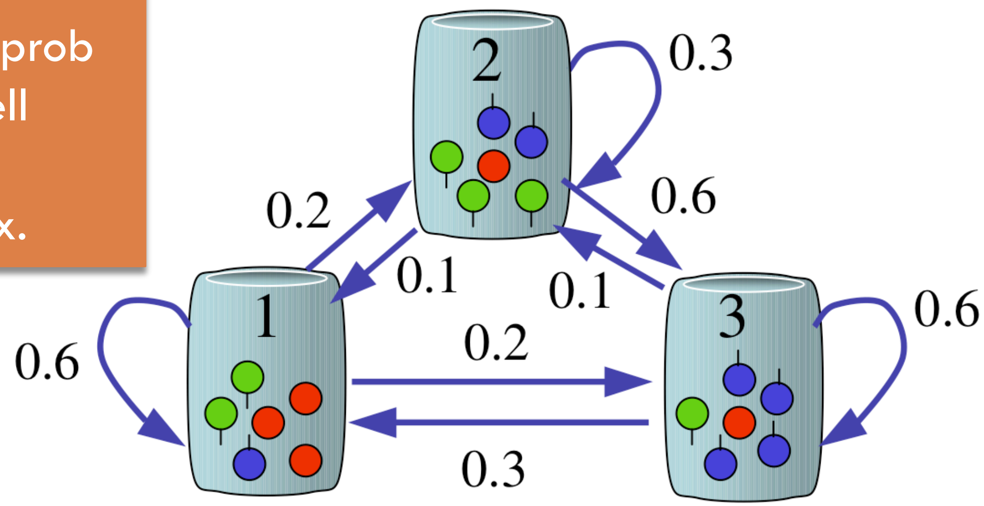
State\Symbol	R	R	G	B
State = 1	(1/3)*(3/6)=1/6	1/20	??	
State = 2	(1/3)*(1/6)=1/18	?		
State = 3	(1/3)*(1/6)=1/18	???		

e.g., for the cell (State = 1, 2nd R), it considers:

1. Prev state is 1: prob = (1/6)*0.6*(3/6)
2. Prev state is 2: prob = (1/18)*0.1*(3/6)
3. Prev state is 3: prob = (1/18)*0.3*(3/6)

Max of the three options is the first one with prob value of (1/20), hence the value in the cell (and $\delta[O_1 \rightarrow q_0]$)

Also "remembers" which prev state is the max.



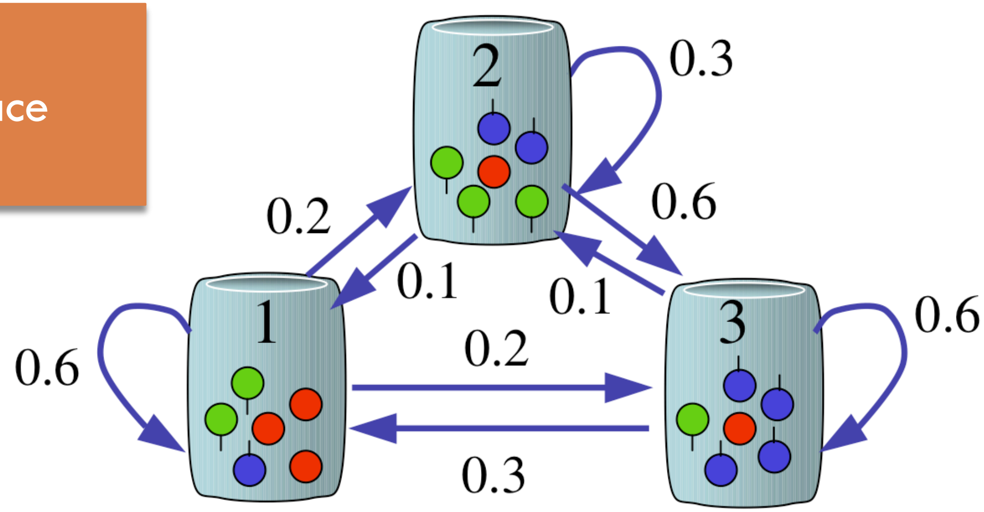
Example of Viterbi Algorithm

State\Symbol	R	R	G	B
State = 1	(1/3)*(3/6)=1/6	1/20	1/100	1/1000
State = 2	(1/3)*(1/6)=1/18	1/180	1/200	1/1500
State = 3	(1/3)*(1/6)=1/18	1/180 (all)	1/600	1/500

Tracing back, we know the best state sequence in terms of the generative probability of the observed symbol sequence is RRGB

Time complexity: $O(T*N^2)$
Space complexity: $O(T*N)$, as we need to trace back

All computation of probabilities should be performed in the log space to avoid underflow. E.g., log(p1*p2) = log(p1) + log(p2)



A Brief Introduction to Proj1

- Input:
  - An HMM model
  - A test file; each line is an address to be parsed
- Output:
  - Top-k parsed results (i.e., state sequences) and their corresponding log-probability score
- Notes
  - Special states: BEGIN and END
  - Add-1 smoothing
  - Tokenization of the address line

Address Parsing Example

- States: {BEGIN, ST#, STNM, STTYP, CITY, STATE, PSTCD}
- Symbols: strings from [0-9a-z]
- Observed symbol sequence:
  - begin 221 Anzac Parade Kingsford NSW 2032 end

    begin ST# STNM STTYP CITY STATE PSTCD end
  - What's the most likely state sequence?
- Enables us to perform advanced tasks, such as deduplication and advanced queries
  - begin 10 Kingsford St, Fairy Meadow, NSW 2519 end

    STNM

Smoothing

- Emission probabilities
  - Let Sybmols = {a, b, c, d, …, z}
  - Without smoothing, Pr[S➔x] = #(S,x) / #(S)
  - Hence if #(S, x) = 0, the probability is 0.
  - With add-1 smoothing, Pr[S➔x] = [#(S,x)+1] / [#(S)+ |Symbols|+1]
    - Denominator needs +1 for Out-of-vocabulary (OOV) sybmol
- State transition probabilities
  - With add-1 smoothing, Pr[S_1➔S_2] = [#(S_1,S_2)+1] / [#(S_1)+|States|]
- Special procedure to handle the BEGIN/END states (and its impact)

References

- Section 5 in “A Revealing Introduction to Hidden Markov Models” by Mark Stamp.
- Sung-jung Cho, “Introduction to Hidden Markov Model and Its Application”
- Ankur Jain, “Hidden Markov Models”