Language and Computation

week 6, Tuesday, February 18, 2014

Tamás Biró
Yale University
tamas.biro@yale.edu
http://www.birot.hu/courses/2014-LC/

Practical matters

- **Post-reading:**
  Chapter 4: intro, 4.1-4.3, 4.5.intro, 4.5.1, 4.5.2, 4.8, 4.12.
  Chapter 5: intro, 5.1-5.4 and more to come.
  Chapter 9: only at the depth discussed in class.

- **Python:** H 6-10, especially `re` in Chapt. 10.

- **Sections:** Python NLTK
  Bird, Klein, Loper: *Natural Language Processing with Python*, Ch 1, `http://www.nltk.org/book/ch01.html`

Today

- Part-of-speech (POS) tagging
- Basics of automatic speech recognition (ASR)
- The idea of *Bayesian inference*
- Markov chains – parameter estimation
- Markov models – three problems (Ferguson)
- The Viterbi Algorithm

From n-grams models to Markov chains

Part-of-Speech Tagging

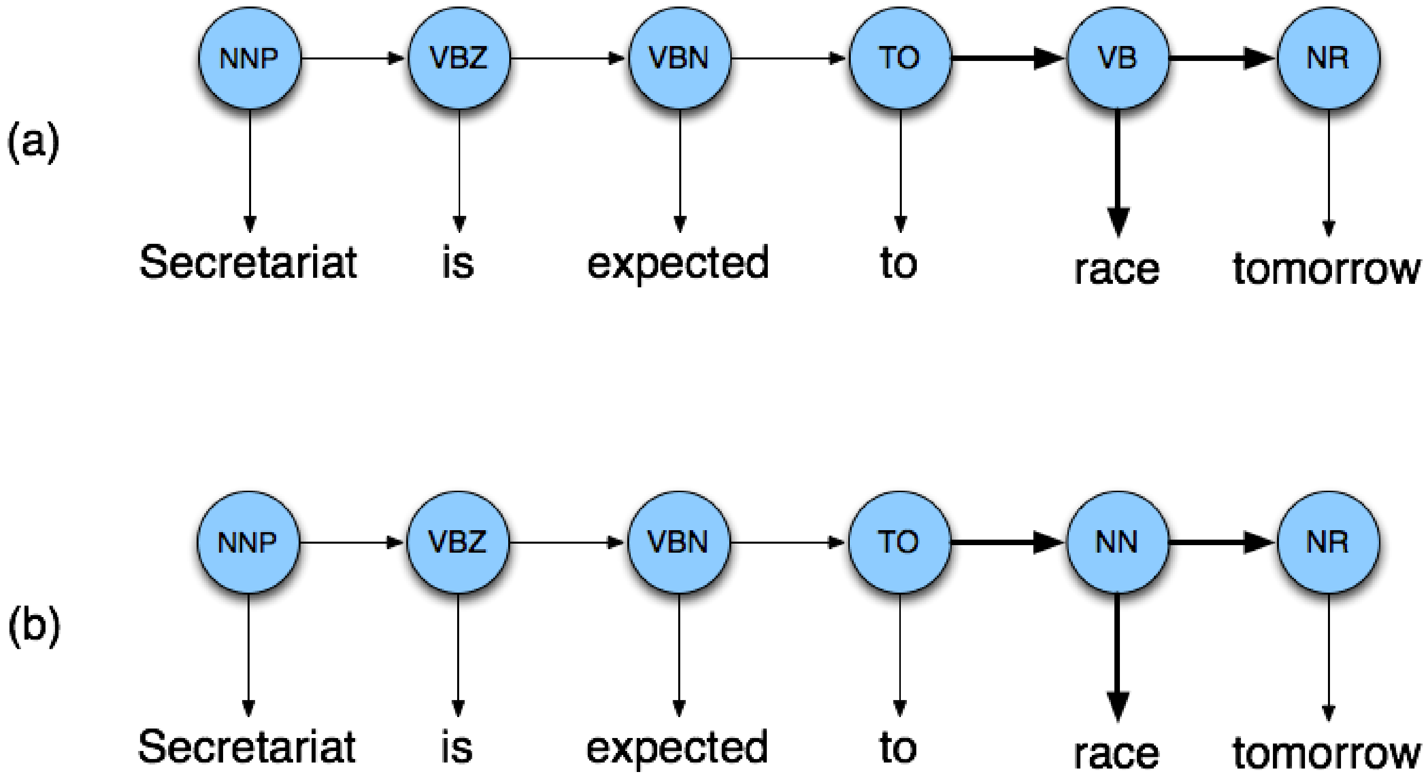
Bag-of-words model: syntax ignored

- Theoretical syntax:
  - trees, context free grammars (next week)
  - more powerful formalisms (cf. *Formal Foundations* course)
  - HPSG and unification (cf. Chapter 15, not in this course)

- Theoretical syntax + probability: PCFG (in two weeks time).

- A useful approximation: Markov Chains

Part-of-Speech Tagging



Bayesian inference

- Bayes' theorem:

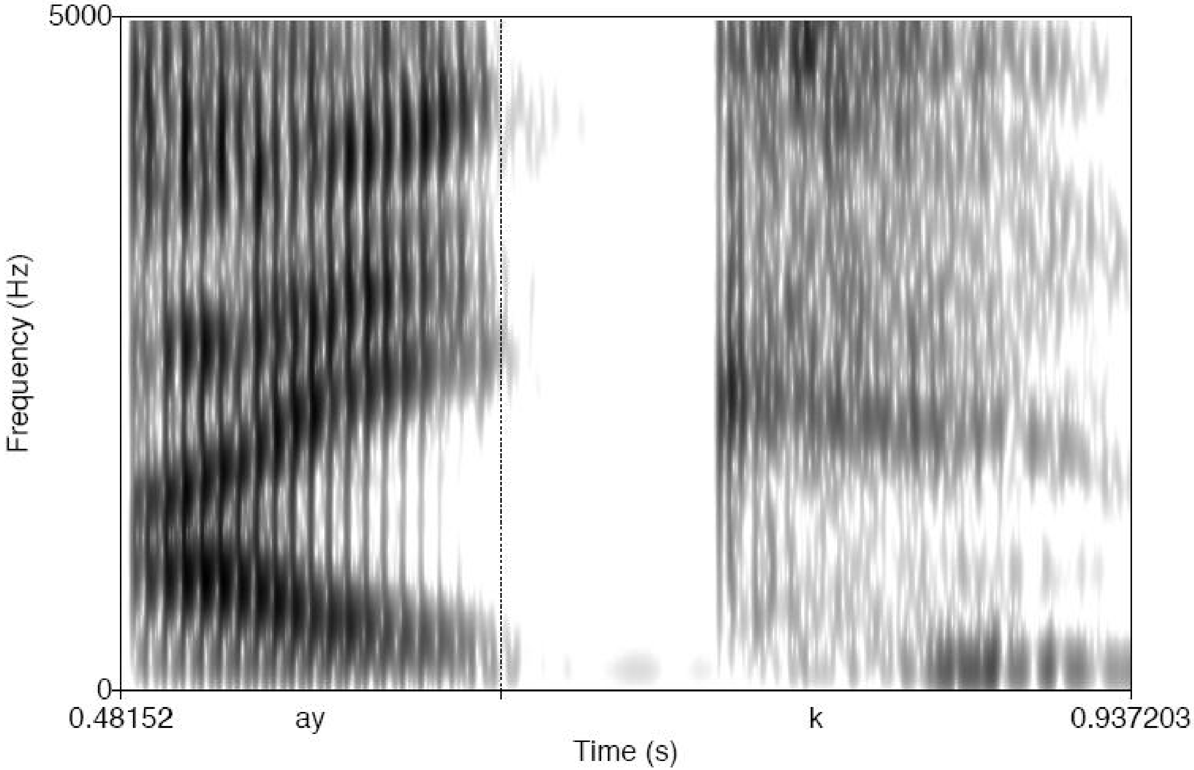
$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

- Example: $A =$ set of POS-tags, $B =$ observation.

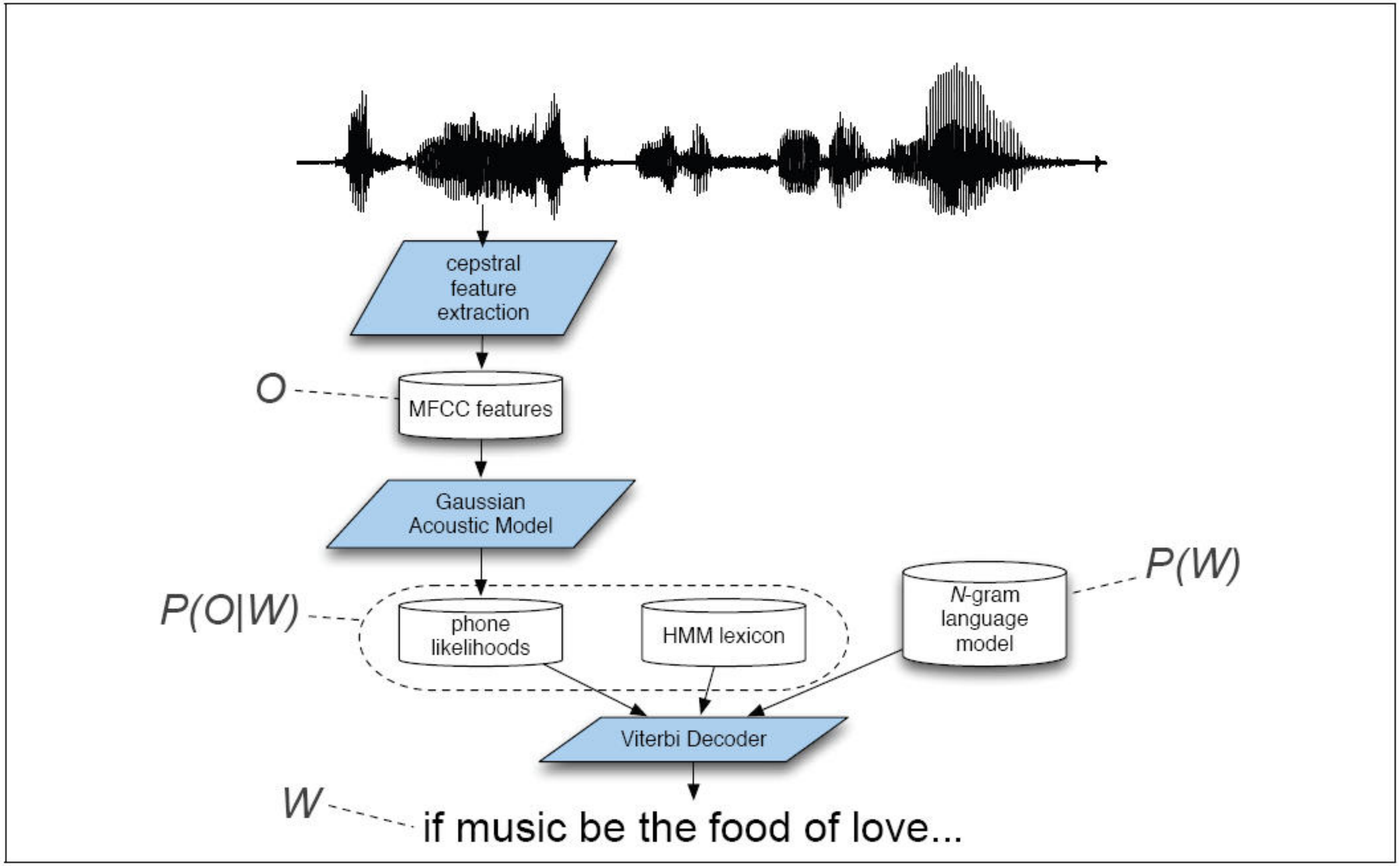
$$\arg\max_A P(A|B) = \arg\max_A (P(B|A) \cdot P(A))$$

- $P(A) =$ **prior probabilities**. $P(B|A) =$ **likelihood**.

Speech recognition



Speech recognition

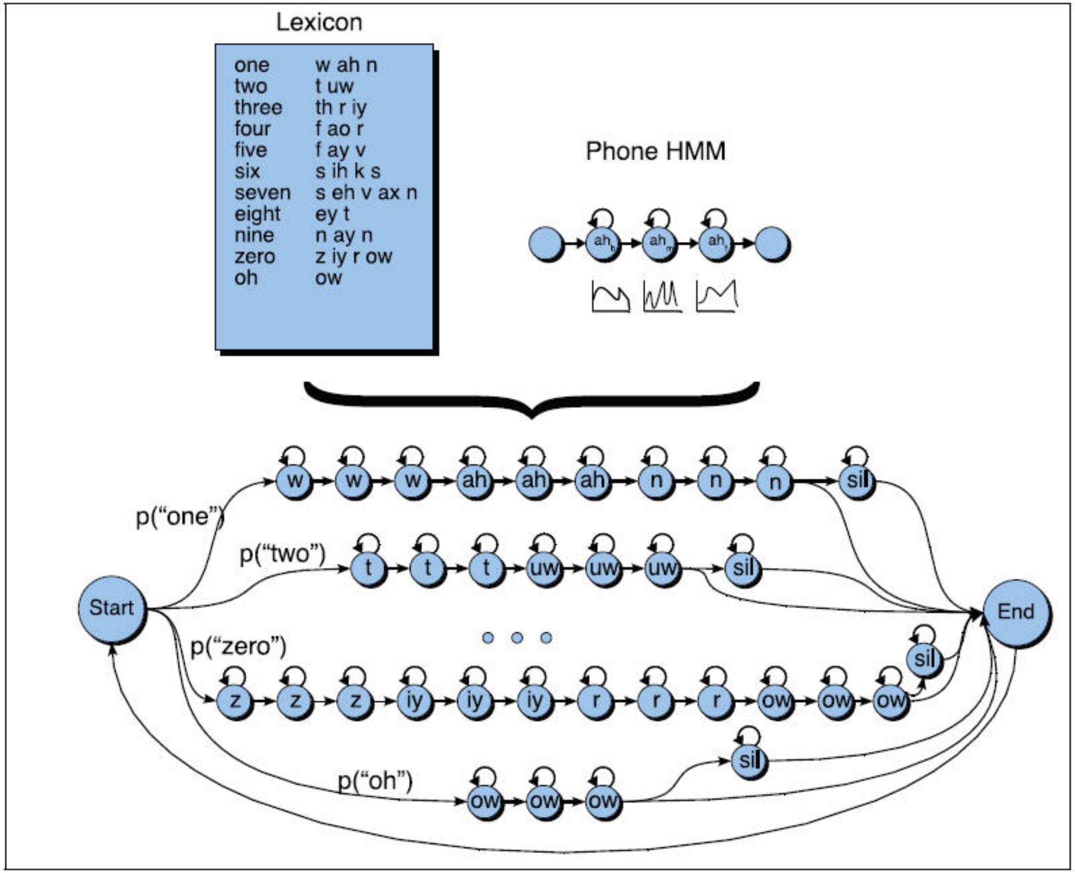


Speech recognition

12	cepstral coefficients
12	delta cepstral coefficients
12	double delta cepstral coefficients
1	energy coefficient
1	delta energy coefficient
1	double delta energy coefficient
39	MFCC features

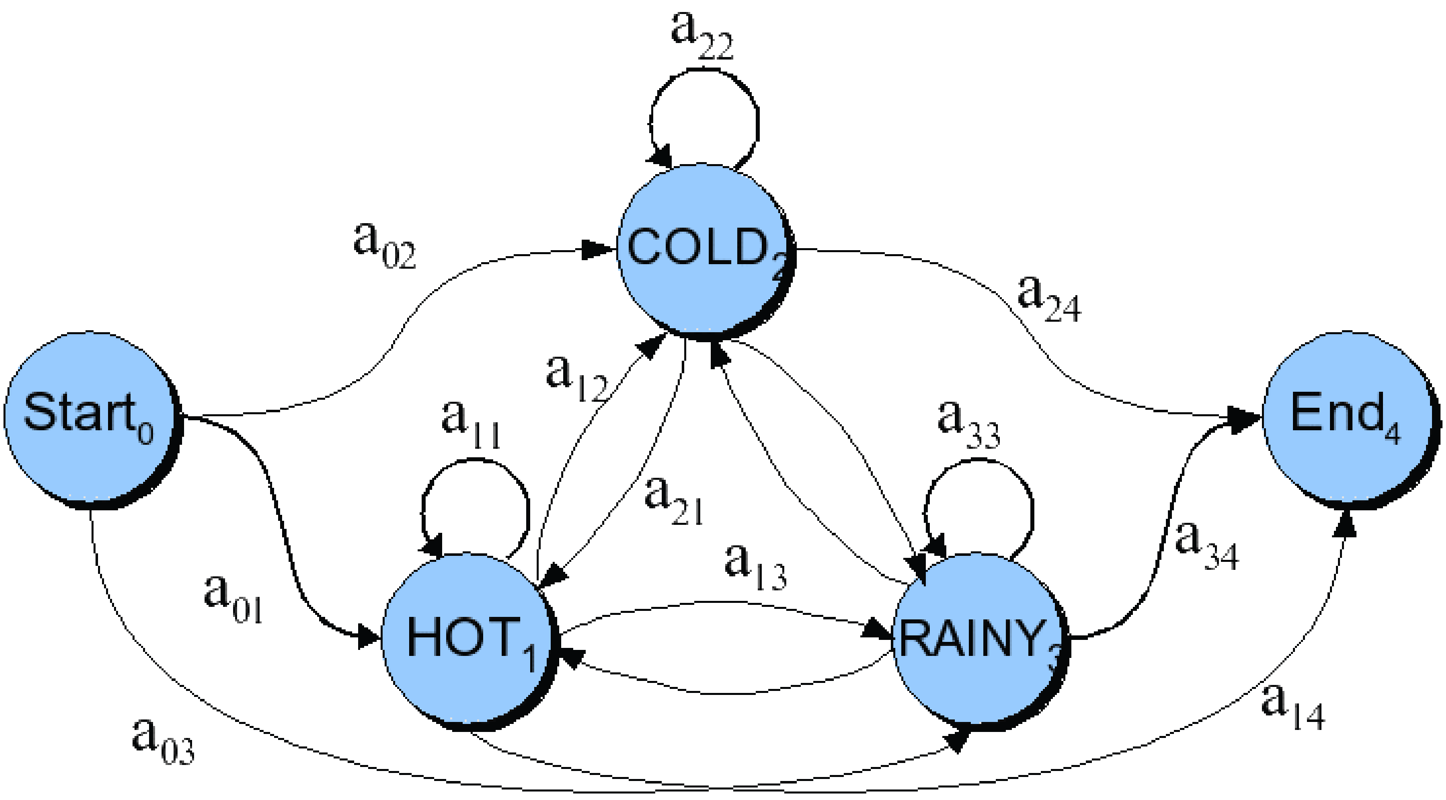
Chapter 9:
page through it to get an idea of the technical details.

Speech recognition

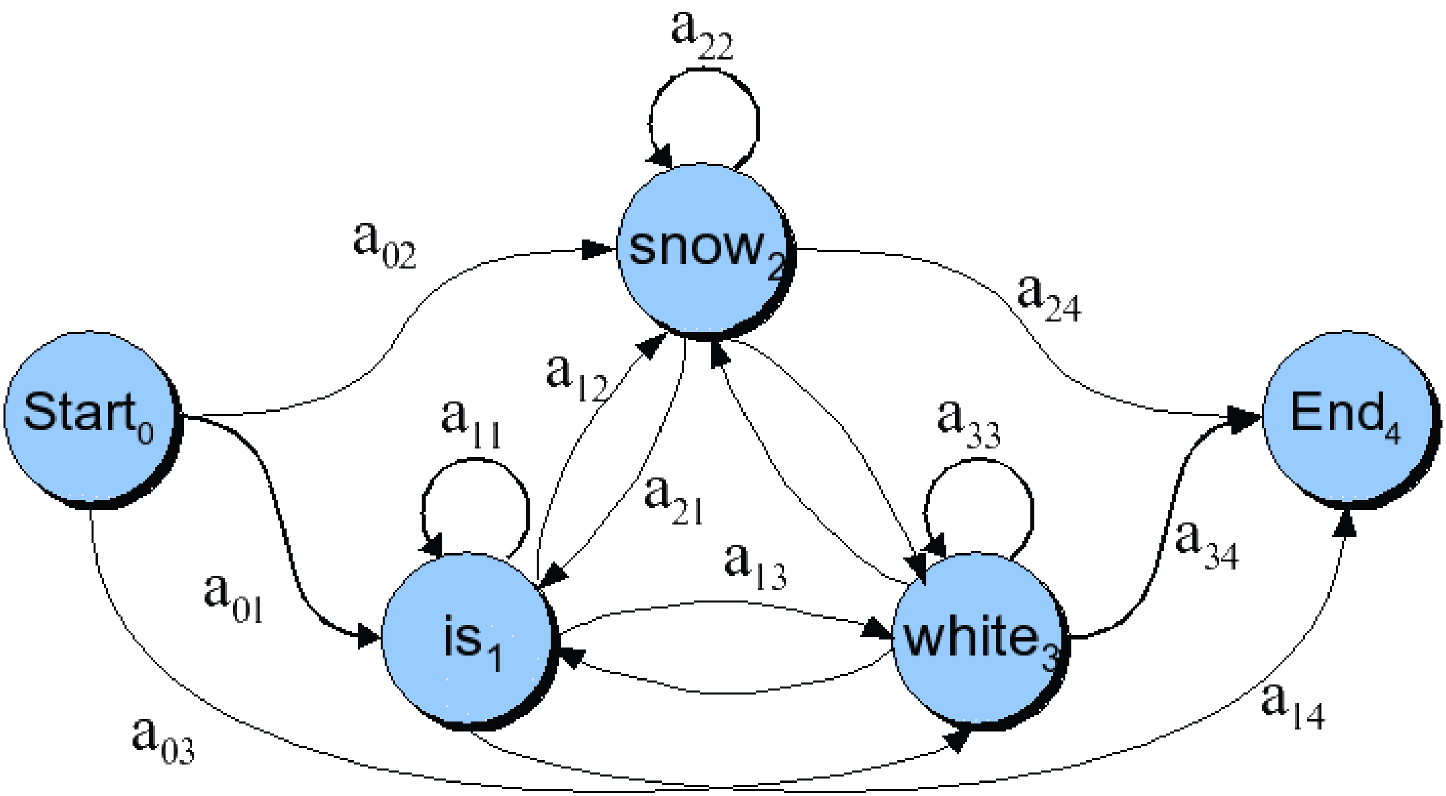


Markov Chains

Markov Chain for Weather



Markov Chain for Words



Markov Chain:
"First-order observable Markov Model"

- Set of states Q. The state at time t is q_t.
- a_{ij}: probability transitioning $q_i \rightarrow q_j$.
- Transition matrix $A = (a_{ij})$.
- Current state depends **only** on previous state:

$$P(q_i|q_1 \ldots q_{i-1}) = P(q_i|q_{i-1}) = a_{i-1,i}$$

Markov Chain:
"First-order observable Markov Model"

- Given Markov Chain, generate a string: trivial.
- Given string, learn a Markov Model:
  - $Q =$ observation types.
  - $a_{i,j} = P(q_j|q_i) =?$
  - *Maximum Likelihood Estimate*:

$$a_{i,j} = P(q_j|q_i) = \frac{P(q_i q_j)}{P(q_i)} = \frac{\#\text{ of } q_i q_j \text{ bigrams}}{\#\text{ of } q_i \text{ unigrams}}$$

  - Laplace Smoothing, Good-Turing Discounting, interpolation, backoff.

Markov Models

Probabilistic Finite State Automaton

Add probability to transitions:

- A quintuple $(Q, \Sigma, q_0, F, \delta(q,i))$
- $\delta(q,i)$ is
  - $\in Q$ for a **deterministic FSA**
  - $\subseteq Q$ for a **non-deterministic FSA**
  - a probability distribution over Q for a **probabilistic FSA**
- When in state q_j and read character i from input tape: move to state q_k with probability $\delta(q_k, i)[q_k]$, for all $q_k \in Q$.

Markov Models: sextuple $(Q, \Sigma, q_0, Q_F, A, B)$

Slightly different terminology, slightly different idea.

- Q finite set of states $q_1, q_2, \ldots, q_N$.
  Σ set of possible observations (finite? not finite?)

- q_0 start state (or probability distribution π over Q)
  q_F end (final) state (or $F \subseteq Q$?)

- A **transition probability matrix**: $\forall i : \sum_{j=1}^{N} a_{ij} = 1$

- B **emission probabilities**: $\forall i : \sum_{o \in \Sigma} b_i(o) = 1$

(Hidden) Markov Models

$Q = q_1 q_2 \ldots q_N$	a set of **states**
$A = a_{01} a_{02} \ldots a_{n1} \ldots a_{nn}$	a **transition probability matrix** A, each a_{ij} representing the probability of moving from state i to state j, s.t. $\sum_{j=1}^{n} a_{ij} = 1 \quad \forall i$
$O = o_1 o_2 \ldots o_N$	a set of **observations**, each one drawn from a vocabulary $V = v_1, v_2, ..., v_V$.
$B = b_i(o_t)$	a set 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_{end}	special **start and end states** that are not associated with observations

(Hidden) Markov Models

- Given MM $\lambda = (A, B)$, generate series of observation: trivial.

- Given MM $\lambda = (A, B)$, given observation sequence O determine:
  - likelihood $P(O|\lambda)$: **forward algorithm**
  - find most probable sequence of states: **Viterbi algorithm**

- Given an observation sequence O, learn A and B: **forward-backward algorithm** (aka Baum-Welch algorithm, special case of Expectation-Maximization/EM algorithm).

Viterbi algorithm

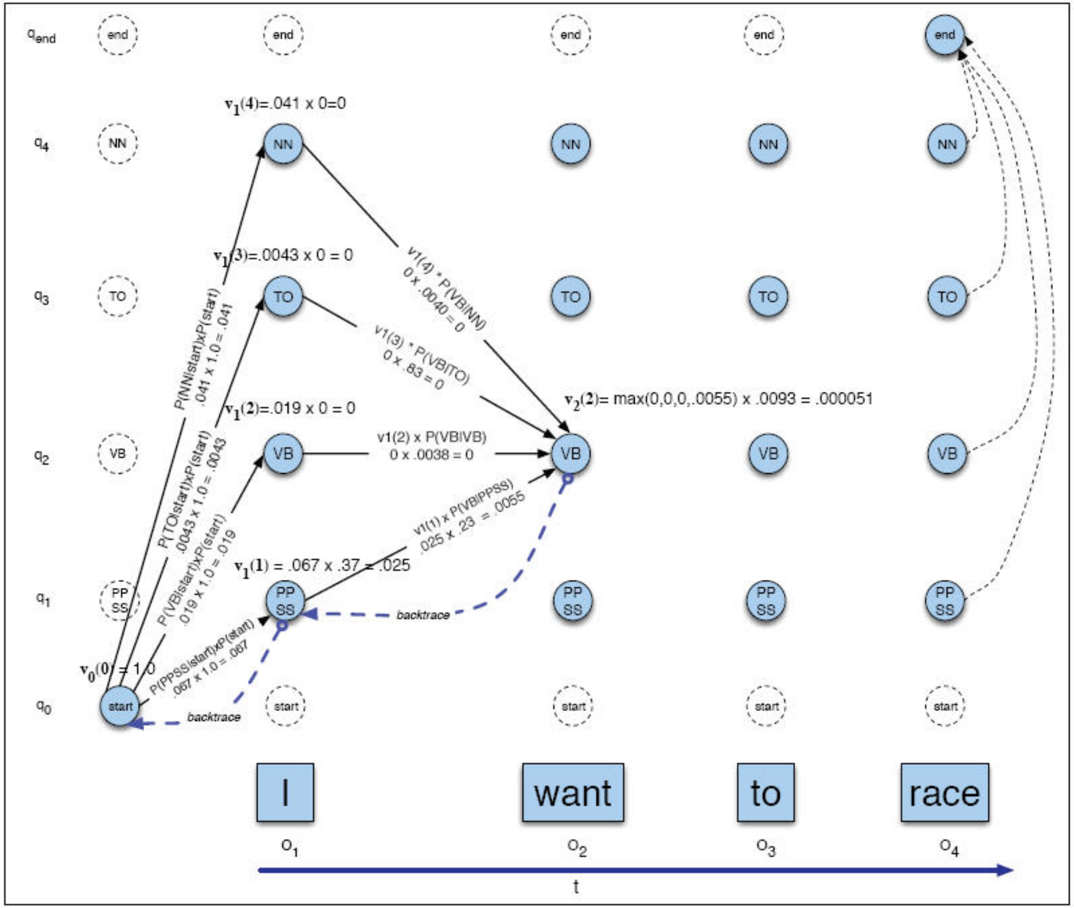
Viterbi algorithm

function VITERBI(*observations* of len T,*state-graph* of len N) **returns** *best-path*

create a path probability matrix *viterbi[N+2,T]*
for each state s **from** 1 **to** N **do** ; initialization step
$viterbi[s,1] \leftarrow a_{0,s} * b_s(o_1)$
$backpointer[s,1] \leftarrow 0$
for each time step t **from** 2 **to** T **do** ; recursion step
for each state s **from** 1 **to** N **do**
$$viterbi[s,t] \leftarrow \max_{s'=1}^{N} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$$
$$backpointer[s,t] \leftarrow \operatorname{argmax}_{s'=1}^{N} viterbi[s',t-1] * a_{s',s}$$
$$viterbi[q_F,T] \leftarrow \max_{s=1}^{N} viterbi[s,T] * a_{s,q_F}$$; termination step
$$backpointer[q_F,T] \leftarrow \operatorname{argmax}_{s=1}^{N} viterbi[s,T] * a_{s,q_F}$$; termination step
return the backtrace path by following backpointers to states back in time from $backpointer[q_F,T]$

Viterbi algorithm

$v_{t-1}(i)$	the **previous Viterbi 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



q_{end}
q_4
q_3
q_2
q_1
q_0
end
NN
TO
VB
PP SS
start
$v_1(4)$=.041 x 0=0
$v_1(3)$=.0043 x 0 = 0
$v_1(2)$=.019 x 0 = 0
$v_1(1)$ = .067 x .37 = .025
$v_0(0)$ = 1.0
P(NN|start)xP(start) .041 x 1.0 = .041
P(TO|start)xP(start) .0043 x 1.0 = .0043
P(VB|start)xP(start) .019 x 1.0 = .019
P(PPSS|start)xP(start) .067 x 1.0 = .067
v1(4) * P(VB|NN) 0 x .0040 = 0
v1(3) * P(VB|TO) 0 x .83 = 0
v1(2) x P(VB|VB) 0 x .0038 = 0
v1(1) x P(VB|PPSS) .025 x .23 = .0055
$v_2(2)$= max(0,0,0,.0055) x .0093 = .000051
backtrace
backtrace
I
want
to
race
o_1
o_2
o_3
o_4
t

See you on Thursday!