CS 481

Artificial Intelligence Language Understanding

February 9, 2023

Announcements / Reminders

- Please follow the Week 05 To Do List instructions
- Quiz #04 due on Sunday (02/12/23) at 11:59 PM CST
- PA #01 due on Monday (02/20/23) at 11:59 PM CST

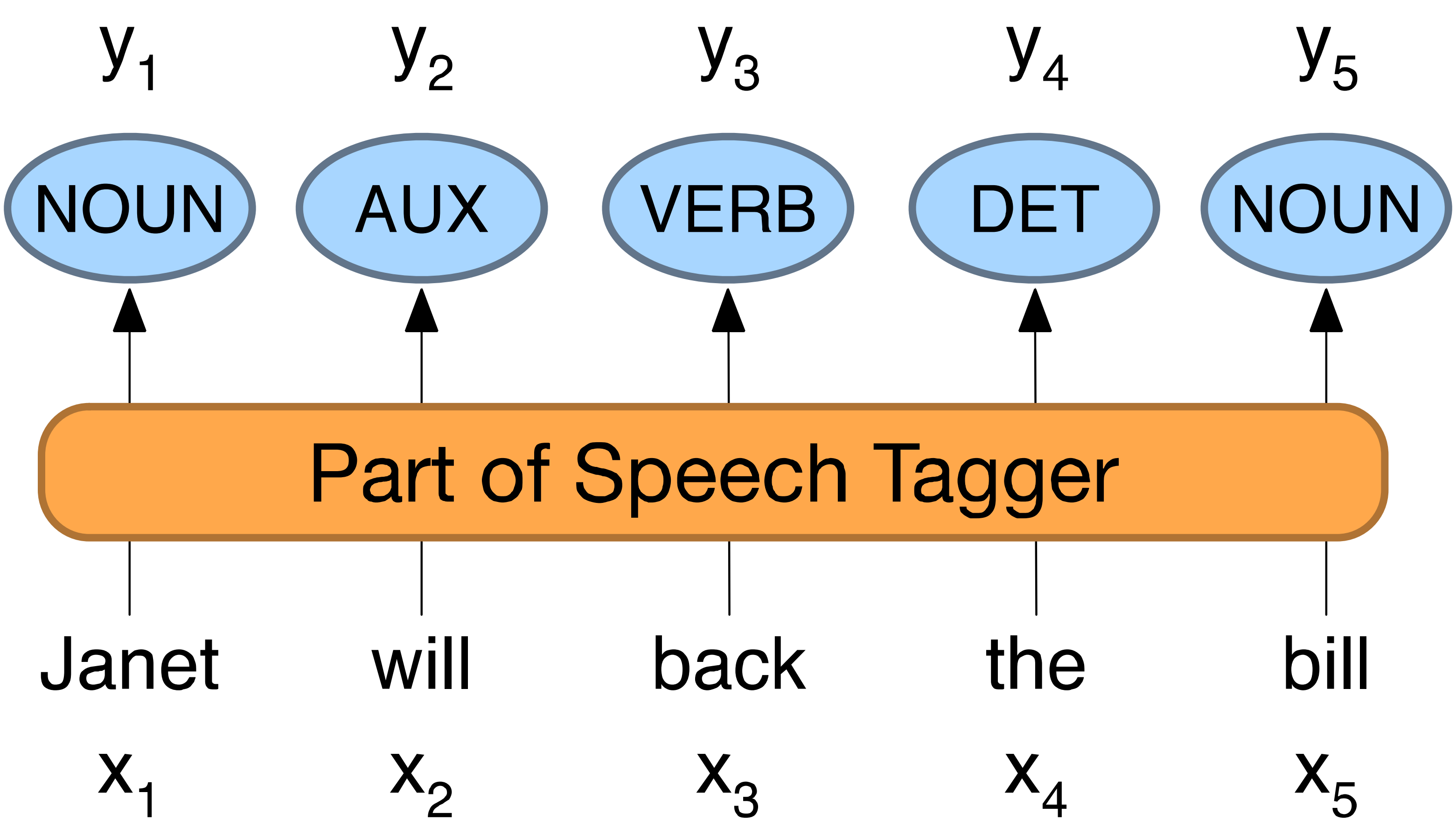
- Exam dates:
  - Midterm: 03/02/2023 during Thursday lecture time
  - Final: 04/27/2023 during Thursday lecture time

Plan for Today

- **Parts of Speech tagging - continued**

Part of Speech Tagging

- **Task:**
  - **Map sequence $x_1,\ldots,x_n$ of words to $y_1,\ldots,y_n$ of POS tags**



Parts of Speech: Tagset Example

Parts of Speech in the Universal Dependencies tagset

	Tag	Description	Example
Open Class	**ADJ**	Adjective: noun modifiers describing properties	*red*, *young*, *awesome*
	ADV	Adverb: verb modifiers of time, place, manner	*very*, *slowly*, *home*, *yesterday*
	NOUN	words for persons, places, things, etc.	*algorithm*, *cat*, *mango*, *beauty*
	VERB	words for actions and processes	*draw*, *provide*, *go*
	PROPN	Proper noun: name of a person, organization, place, etc..	*Regina*, *IBM*, *Colorado*
	INTJ	Interjection: exclamation, greeting, yes/no response, etc.	*oh*, *um*, *yes*, *hello*
Closed Class Words	**ADP**	Adposition (Preposition/Postposition): marks a noun's spacial, temporal, or other relation	*in, on, by, under*
	AUX	Auxiliary: helping verb marking tense, aspect, mood, etc.,	*can, may, should, are*
	CCONJ	Coordinating Conjunction: joins two phrases/clauses	*and, or, but*
	DET	Determiner: marks noun phrase properties	*a, an, the, this*
	NUM	Numeral	*one, two, first, second*
	PART	Particle: a preposition-like form used together with a verb	*up, down, on, off, in, out, at, by*
	PRON	Pronoun: a shorthand for referring to an entity or event	*she, who, I, others*
	SCONJ	Subordinating Conjunction: joins a main clause with a subordinate clause such as a sentential complement	*that*, *which*
Other	**PUNCT**	Punctuation	; , ()
	SYM	Symbols like $ or emoji	$, %
	X	Other	asdf, qwfg

Parts of Speech: Tagset Example

Penn Treebank Parts-of-speech tags:

Tag	Description	Example	Tag	Description	Example	Tag	Description	Example
CC	coord. conj.	*and, but, or*	NNP	proper noun, sing.	*IBM*	TO	"to"	*to*
CD	cardinal number	*one, two*	NNPS	proper noun, plu.	*Carolinas*	UH	interjection	*ah, oops*
DT	determiner	*a, the*	NNS	noun, plural	*llamas*	VB	verb base	*eat*
EX	existential 'there'	*there*	PDT	predeterminer	*all, both*	VBD	verb past tense	*ate*
FW	foreign word	*mea culpa*	POS	possessive ending	*'s*	VBG	verb gerund	*eating*
IN	preposition/ subordin-conj	*of, in, by*	PRP	personal pronoun	*I, you, he*	VBN	verb past partici-ple	*eaten*
JJ	adjective	*yellow*	PRP$	possess. pronoun	*your, one's*	VBP	verb non-3sg-pr	*eat*
JJR	comparative adj	*bigger*	RB	adverb	*quickly*	VBZ	verb 3sg pres	*eats*
JJS	superlative adj	*wildest*	RBR	comparative adv	*faster*	WDT	wh-determ.	*which, that*
LS	list item marker	*1, 2, One*	RBS	superlatv. adv	*fastest*	WP	wh-pronoun	*what, who*
MD	modal	*can, should*	RP	particle	*up, off*	WP$	wh-possess.	*whose*
NN	sing or mass noun	*llama*	SYM	symbol	*+,%, &*	WRB	wh-adverb	*how, where*

Sample Tagged Sentence

There/PRO were/VERB 70/NUM children/NOUN there/ADV ./PUNC

Preliminary/ADJ findings/NOUN were/AUX reported/VERB in/ADP today/NOUN ’s/PART New/PROPN England/PROPN Journal/PROPN of/ADP Medicine/PROPN

Standard POS Tagging Models

- Supervised Machine Learning Algorithms:
  - Hidden Markov Models
  - Conditional Random Fields (CRF) / Maximum Entropy Markov Models (MEMM)
  - Neural sequence models (RNNs or Transformers)
  - Large Language Models (like BERT), finetuned
- All required a hand-labeled training set, all about equal performance (97% on English)
  - All make use of information sources we discussed
  - Via human created features: HMMs and CRFs
  - Via representation learning: Neural LMs

Part of Speech: Conditional Probability

$$P(Category = NOUN \mid word\ =\ flies) = \frac{P(word\ =\ flies, Category = NOUN)}{P(Word\ = flies)}$$

where $\mathrm{P}(Word\ =\ flies) > 0$

Part of Speech: Conditional Probability

$$P(C = NOUN \mid w = flies) = \frac{P(w = flies, C = NOUN)}{P(w = flies)}$$

where $\mathrm{P}(w = flies) > 0$

Most Frequent Class Tagging

With:

$$P(w = flies) = 1000/1\,273\,000 = 0.0008$$

$$P(w = flies, C = NOUN) = 400/1\,273\,000 = 0.0003$$

$$P(w = flies, C = VERB) = 600/1\,273\,000 = 0.0005$$

$$P(C = NOUN \mid w = flies) = \frac{P(w = flies, C = NOUN)}{P(w = flies)} = \frac{0.0003}{0.0008}$$

vs.

$$P(C = VERB \mid w = flies) = \frac{P(w = flies, C = VERB)}{P(w = flies)} = \frac{0.0005}{0.0008}$$

With this approach *flies* will <u>ALWAYS</u> be tagged as a VERB.

Bayes' Rule

$$P(A \mid B) = \frac{P(B \mid A) * P(A)}{P(B)}$$

POS Tagging: General Approach

Given a sequence of words (a "sentence"):

$$w_1, w_2, w_3, \ldots, w_T$$

there is going to be a corresponding sequence of lexical categories:

$$C_1, C_2, C_3, \ldots, C_T$$

What is most likely sequence of categories?

POS Tagging: General Approach

To answer this we would want to find a conditional probability:

$$P(C_1, C_2, C_3, \ldots, C_T \mid w_1, w_2, w_3, \ldots, w_T)$$

In other words: what is the probability of having a sequence of lexical categories

$$C_1, C_2, C_3, \ldots, C_T$$

GIVEN that the sequence of words is

$$w_1, w_2, w_3, \ldots, w_T\ ?$$

POS Tagging: General Approach

The probability we are looking for

$$P(C_1, C_2, C_3, \ldots, C_T \mid w_1, w_2, w_3, \ldots, w_T)$$

will require a lot of data, which we most likely won't have.

We can use Bayes' Theorem:

$$\boldsymbol{P(C_1, C_2, C_3, \ldots, C_T \mid w_1, w_2, w_3, \ldots, w_T) =}$$

$$\boldsymbol{= \frac{P(w_1, w_2, w_3, \ldots, w_T \mid C_1, C_2, C_3, \ldots, C_T) * P(C_1, C_2, C_3, \ldots, C_T)}{P(w_1, w_2, w_3, \ldots, w_T)}}$$

POS Tagging: General Approach

In order to find the most likely sequence:

$$C_1, C_2, C_3, ..., C_T$$

we need to maximize (most likely sequence!):

$$P(C_1, C_2, C_3, \ldots, C_T \mid w_1, w_2, w_3, \ldots, w_T) =$$

$$= \frac{P(w_1, w_2, w_3, \ldots, w_T \mid C_1, C_2, C_3, \ldots, C_T) * (C_1, C_2, C_3, \ldots, C_T)}{P(w_1, w_2, w_3, \ldots, w_T)}$$

POS Tagging: General Approach

Maximizing :

$$P(C_1, C_2, C_3, ..., C_T \mid w_1, w_2, w_3, ..., w_T)$$

in practice means maximizing the numerator:

$$\frac{P(w_1, w_2, w_3, ..., w_T \mid C_1, C_2, C_3, ..., C_T) * P(C_1, C_2, C_3, ..., C_T)}{P(w_1, w_2, w_3, ..., w_T)}$$

as denominator $P(w_1, w_2, w_3, ..., w_T)$ will not change:

POS Tagging: General Approach

Estimating:

$$P(w_1, w_2, w_3, \ldots, w_T \mid C_1, C_2, C_3, \ldots, C_T) * P(C_1, C_2, C_3, \ldots, C_T)$$

using counts once again requires a lot of data that we will likely not have.

Alternative: approximate it with N-grams (here bigrams):

$$P(C_1, C_2, C_3, \ldots, C_T) = \prod_{i=1}^{T} P(C_i \mid all\ categories\ preceding\ C_i)$$

$$P(C_1, C_2, C_3, \ldots, C_T) \cong \prod_{i=1}^{T} P(C_i \mid C_{i-})$$

POS Tagging: General Approach

Estimating:

$$P(w_1, w_2, w_3, ..., w_T \mid C_1, C_2, C_3, ..., C_T) * P(C_1, C_2, C_3, ..., C_T)$$

Approximate it with N-grams (here bigrams):

$$P(C_1, C_2, C_3, ..., C_T) = \prod_{i=1}^{T} P(C_i \mid all\ categories\ preceding\ C_i)$$

$$P(C_1, C_2, C_3, ..., C_T) \cong \prod_{i=1}^{T} P(C_i \mid C_{i-1})$$

$$P(w_1, w_2, w_3, ..., w_T \mid C_1, C_2, C_3, ..., C_T) \cong \prod_{i=1}^{T} P(w_i \mid C_i)$$

POS Tagging: General Approach

With approximations:

$$P(w_1, w_2, w_3, ..., w_T \mid C_1, C_2, C_3, ..., C_T) * P(C_1, C_2, C_3, ..., C_T) \cong$$

$$\cong \prod_{i=1}^{T} P(w_i \mid C_i) * P(C_i \mid C_{i-1})$$

and we want to maximize:

$$\prod_{i=1}^{T} P(w_i \mid C_i) * P(C_i \mid C_{i-1})$$

Individual probabilities can now be estimated using corpus counts!

POS Tagging: Simple Tagset

Let's assume we have a simple tagset:

- **N - NOUN**
- **V - VERB**
- **ART - ARTICLE**
- **P - PREPOSITION**

and a some synthetic corpus.

POS Tagging: General Approach

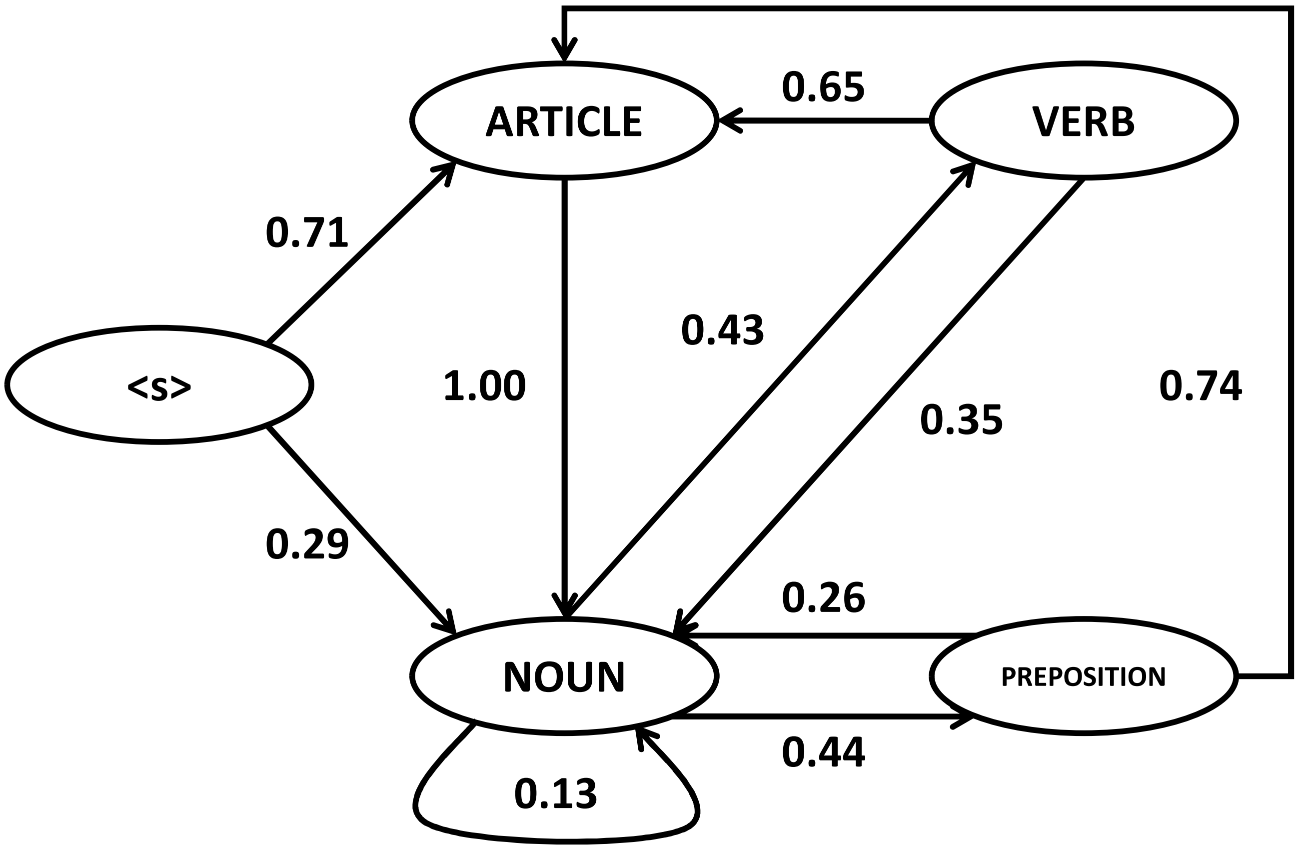
Estimations with corpus counts:

$$P(C_i = VERB \mid C_{i-1} = NOUN) = \frac{Count\ (NOUN\ at\ position\ i -\ \ and\ VERB\ at\ i)}{Count(NOUN\ at\ position\ i-)}$$

Sample bigram probabilities from our synthetic corpus:

Category	Count at i	Pair	Count at i,i+1	P(Bigram)	Estimate
<s>	300	<s>, ARTICLE	213	P(ARTICLE|<S>)	0.71
<s>	300	<s>, NOUN	87	P(NOUN|<S>)	0.29
ARTICLE	558	ARTICLE, NOUN	558	P(NOUN|ARTICLE)	1.00
NOUN	833	NOUN, VERB	358	P(VERB|NOUN)	0.43
NOUN	833	NOUN, NOUN	108	P(NOUN|NOUN)	0.13
NOUN	833	NOUN, PREPOSITION	366	P(PREPOSITION|NOUN)	0.44
VERB	300	VERB, NOUN	75	P(NOUN|VERB)	0.35
VERB	300	VERB, ARTICLE	194	P(ARTICLE|VERB)	0.65
PREPOSITION	307	PREPOSITION, ARTICLE	226	P(ARTICLE|PREPOSITION)	0.74
PREPOSITION	307	PREPOSITION, NOUN	81	P(NOUN|PREPOSITION)	0.26

Hidden Markov Model



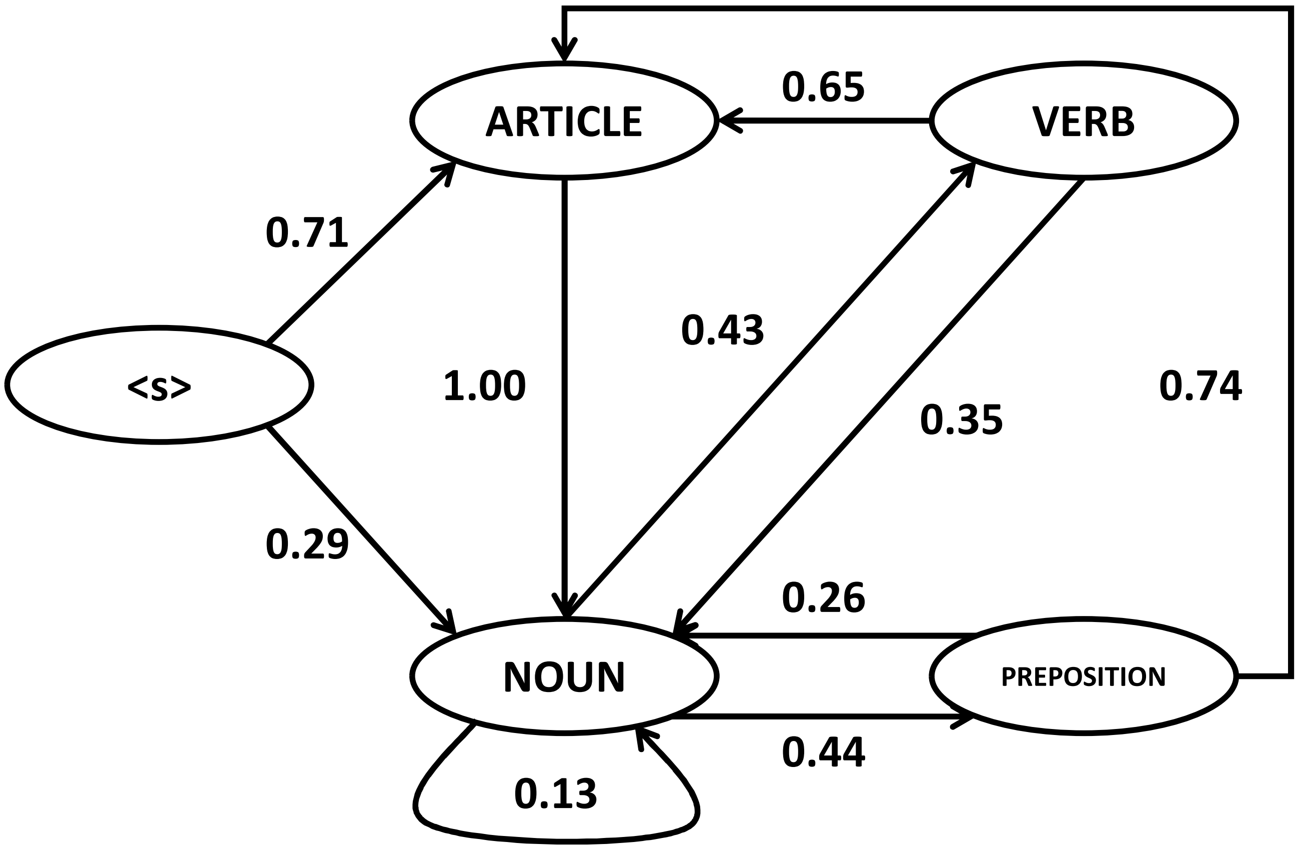
P(Bigram)	Estimate
P(ARTICLE|<S>)	0.71
P(NOUN|<S>)	0.29
P(NOUN|ARTICLE)	1.00
P(VERB|NOUN)	0.43
P(NOUN|NOUN)	0.13
P(PREPOSITION|NOUN)	0.44
P(NOUN|VERB)	0.35
P(ARTICLE|VERB)	0.65
P(ARTICLE|PREPOSITION)	0.74
P(NOUN|PREPOSITION)	0.26

Consider a following sequence of categories (tags):

<s>, ARTICLE, NOUN, VERB, NOUN

What's the probability of its occurence in our synthetic corpus?

Hidden Markov Model (HMM)

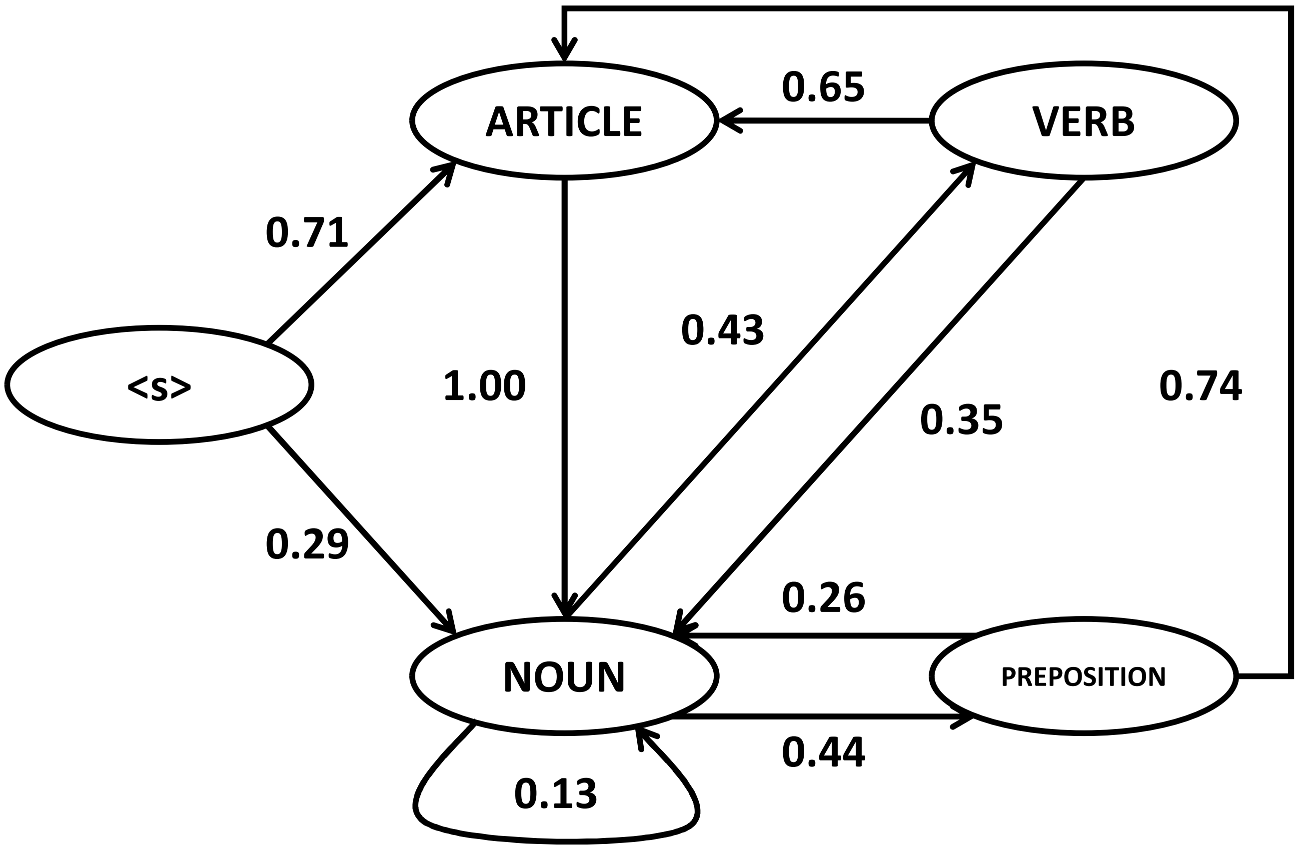


P(Bigram)	Estimate
P(ARTICLE|<S>)	0.71
P(NOUN|<S>)	0.29
P(NOUN|ARTICLE)	1.00
P(VERB|NOUN)	0.43
P(NOUN|NOUN)	0.13
P(PREPOSITION|NOUN)	0.44
P(NOUN|VERB)	0.35
P(ARTICLE|VERB)	0.65
P(ARTICLE|PREPOSITION)	0.74
P(NOUN|PREPOSITION)	0.26

The word "Hidden" in Hidden Markov Model means that for a specific sequence (of words) it is unclear what state the model is in.

The word *flies* could be generated from state NOUN and state VERB.

Hidden Markov Model

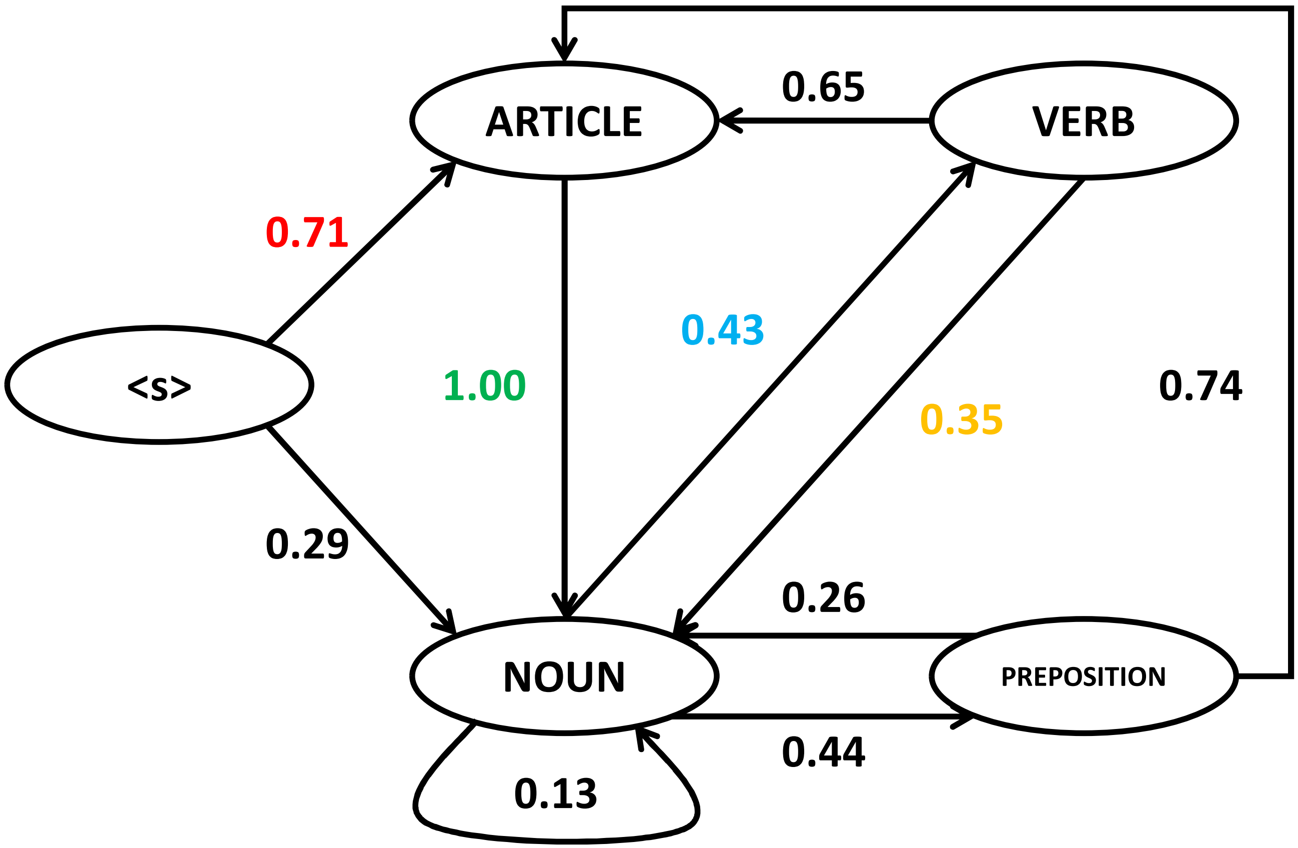


P(Bigram)	Estimate
P(ARTICLE|<S>)	0.71
P(NOUN|<S>)	0.29
P(NOUN|ARTICLE)	1.00
P(VERB|NOUN)	0.43
P(NOUN|NOUN)	0.13
P(PREPOSITION|NOUN)	0.44
P(NOUN|VERB)	0.35
P(ARTICLE|VERB)	0.65
P(ARTICLE|PREPOSITION)	0.74
P(NOUN|PREPOSITION)	0.26

Probability of occurrence of a sequence of categories (tags):

$$P(C_1, C_2, C_3, \ldots, C_T) \cong \prod_{i=1}^{T} P(C_i \mid C_{i-1})$$

Hidden Markov Model



P(Bigram)	Estimate
P(ARTICLE|<S>)	0.71
P(NOUN|<S>)	0.29
P(NOUN|ARTICLE)	1.00
P(VERB|NOUN)	0.43
P(NOUN|NOUN)	0.13
P(PREPOSITION|NOUN)	0.44
P(NOUN|VERB)	0.35
P(ARTICLE|VERB)	0.65
P(ARTICLE|PREPOSITION)	0.74
P(NOUN|PREPOSITION)	0.26

Probability of occurrence of a sequence of categories (tags):

$$P(<s>, \text{ARTICLE, NOUN, VERB, NOUN}) =$$

$$\cong P(ART|<s>) * P(N|ART) * P(V|N) * P(N|V) = 0.71 * 1.00 * 0.43 * 0.35 = 0.107$$

Synthetic Corpus: Word/Tag Counts

Summary of selected word counts in the synthetic corpus:

Word/Tag	N	V	ART	P	TOTAL
flies	21	23	0	0	44
fruit	49	5	1	0	55
like	10	30	0	21	61
a	1	0	201	0	202
the	1	0	300	2	303
flower	53	15	0	0	68
flowers	42	16	0	0	58
birds	64	1	0	0	65
others	592	210	56	284	1142
TOTAL	833	300	558	307	1998

From the table we can calculate lexical generation probabilities $P(w|C)$ estimates:

$P(\textit{the}|ART) = 300/558 = 0.54$

$P(\textit{flies}|N) = 21/833 = 0.025$

$P(\textit{flies}|V) = 23/300 = 0.076$

$P(\textit{like}|V) = 30/300 = 0.1$

$P(\textit{like}|P) = 21/307 = 0.068$

$P(\textit{a}|ART) = 201/558 = 0.36$

$P(\textit{a}|N) = 1/833 = 0.001$

$P(\textit{flower}|N) = 53/833 = 0.063$

$P(\textit{flower}|V) = 15/300 = 0.05$

$P(\textit{like}|N) = 10/833 = 0.012$

Example

Given our synthetic corpus, what is the most like sequence of categories (tags) corresponding to a sentence:

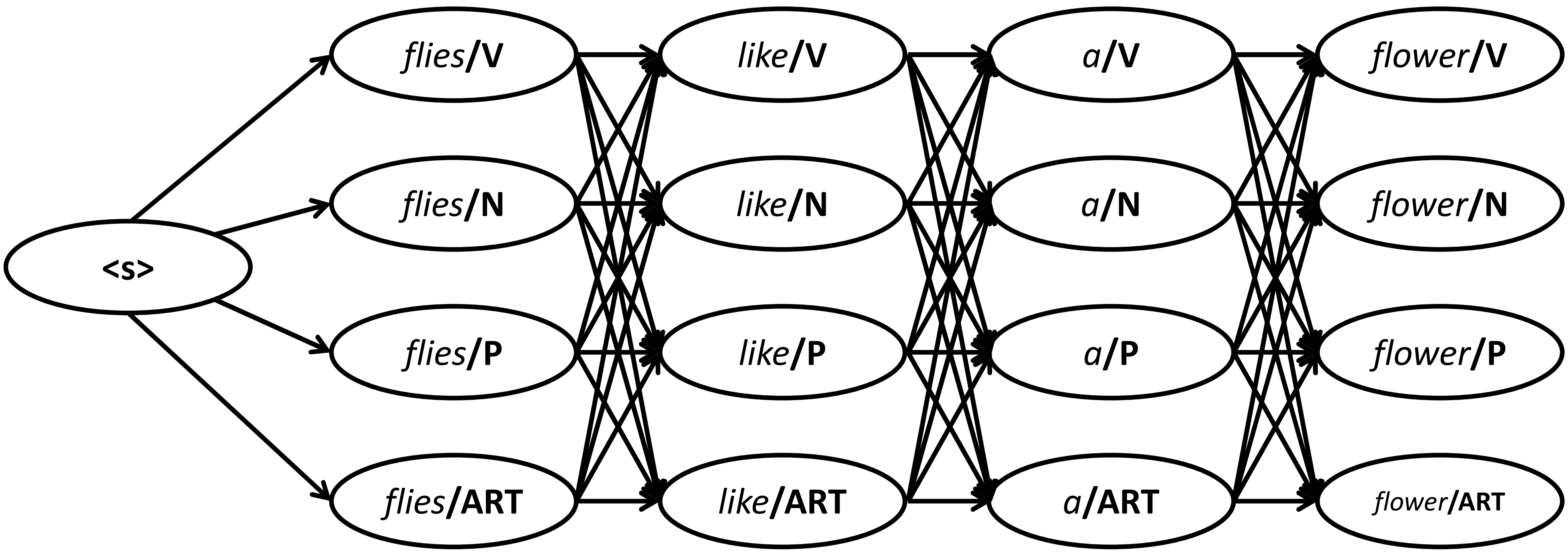
Flies like a flower

We need to maximize:

$$P(w_1, w_2, w_3, \ldots, w_T \mid C_1, C_2, C_3, \ldots, C_T) * P(C_1, C_2, C_3, \ldots, C_T) \cong$$

$$\cong \prod_{i=1}^{T} P(w_i \mid C_i) * P(C_i \mid C_{i-1})$$

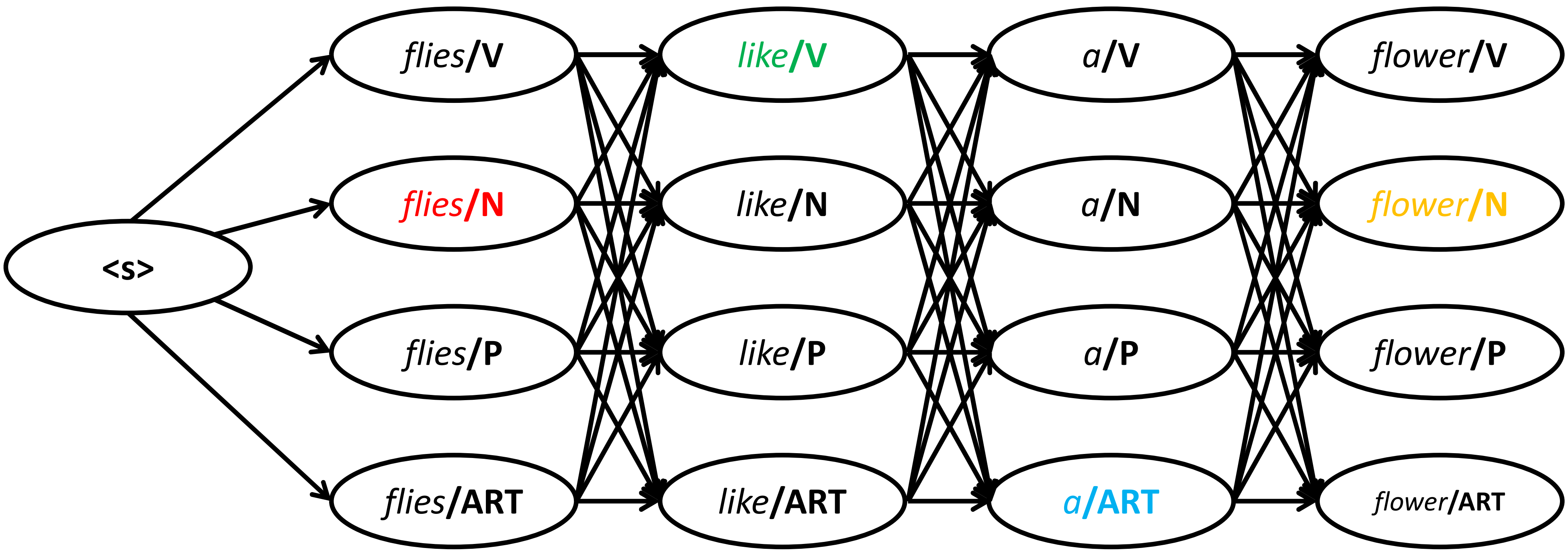
Example: All Possible Sequences



Every sequence can be assigned a probability:

$$P(w_1, w_2, w_3, \ldots, w_T \mid C_1, C_2, C_3, \ldots, C_T) \cong \prod_{i=1}^{T} P(w_i \mid C_i)$$

Example: All Possible Sequences



Every sequence can be assigned a probability:

$$\prod_{i=1}^{T} P(w_i \mid C_i) = P(flies|N) * P(like|V) * P(a|ART) * P(flower|N)$$

Synthetic Corpus: Word/Tag Counts

Summary of selected word counts in the synthetic corpus:

Word/Tag	N	V	ART	P	TOTAL
flies	21	23	0	0	44
fruit	49	5	1	0	55
like	10	30	0	21	61
a	1	0	201	0	202
the	1	0	300	2	303
flower	53	15	0	0	68
flowers	42	16	0	0	58
birds	64	1	0	0	65
others	592	210	56	284	1142
TOTAL	833	300	558	307	1998

From the table we can calculate lexical generation probabilities P(w|C) estimates:

P(*the*|ART) = 300/558 = 0.54

P(*flies*|N) = 21/833 = 0.025

P(*flies*|V) = 23/300 = 0.076

P(*like*|V) = 30/300 = 0.1

P(*like*|P) = 21/307 = 0.068

P(*a*|ART) = 201/558 = 0.36

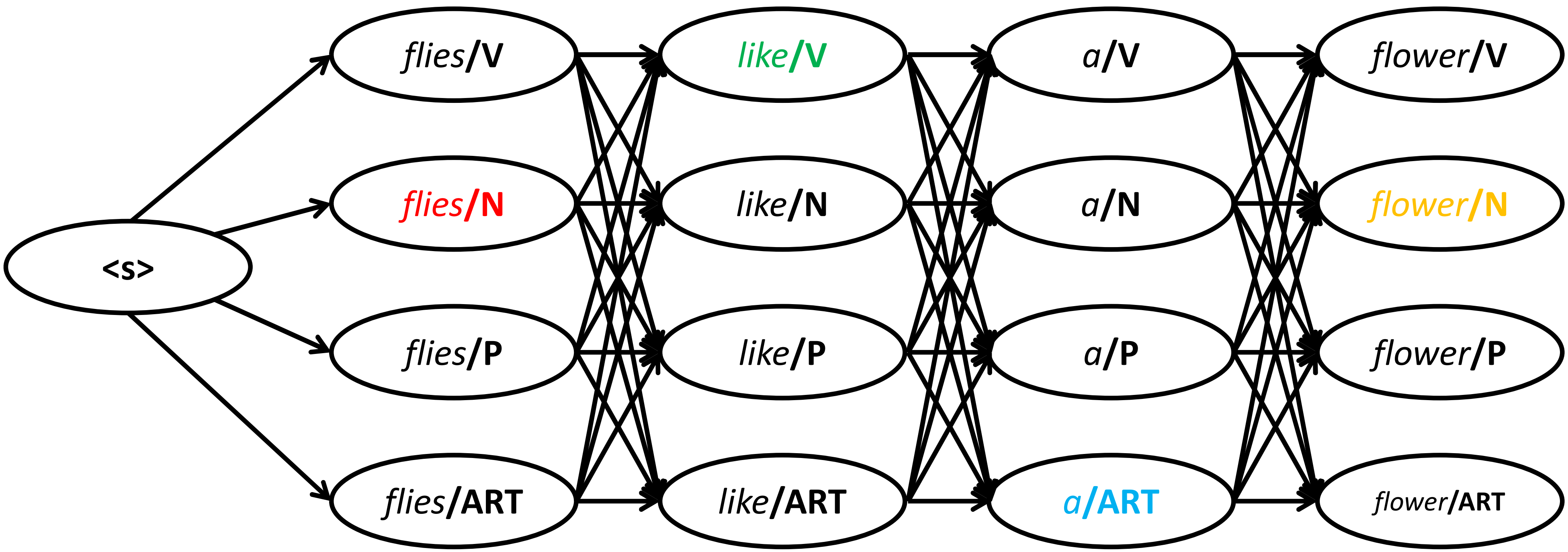
P(*a*|N) = 1/833 = 0.001

P(*flower*|N) = 53/833 = 0.063

P(*flower*|V) = 15/300 = 0.05

P(*like*|N) = 10/833 = 0.012

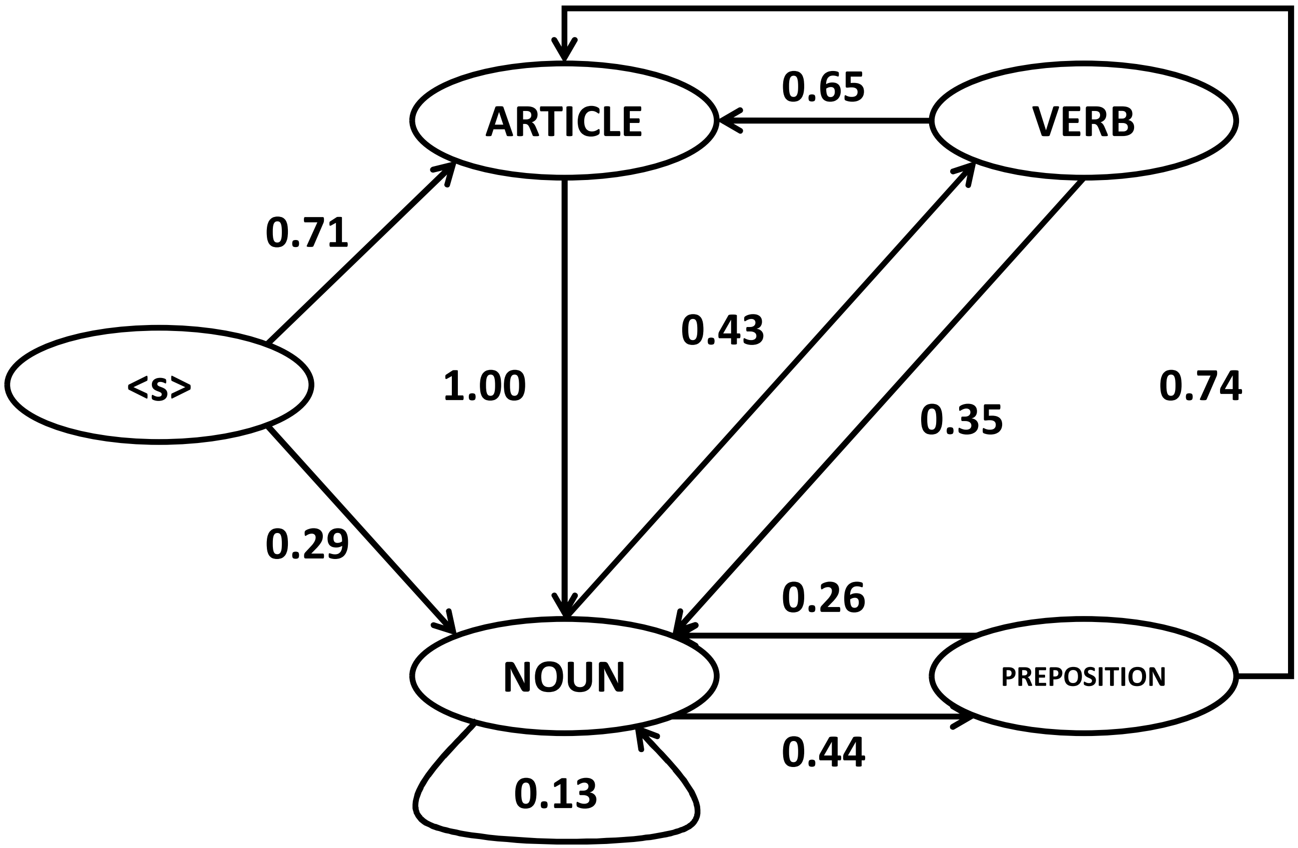
Example: All Possible Sequences



Every sequence can be assigned a probability:

$$\prod_{i=1}^{T} P(w_i \mid C_i) = 0.025 * 0.1 * 0.36 * 0.063 = 5.4 * 0^{-}$$

Hidden Markov Model

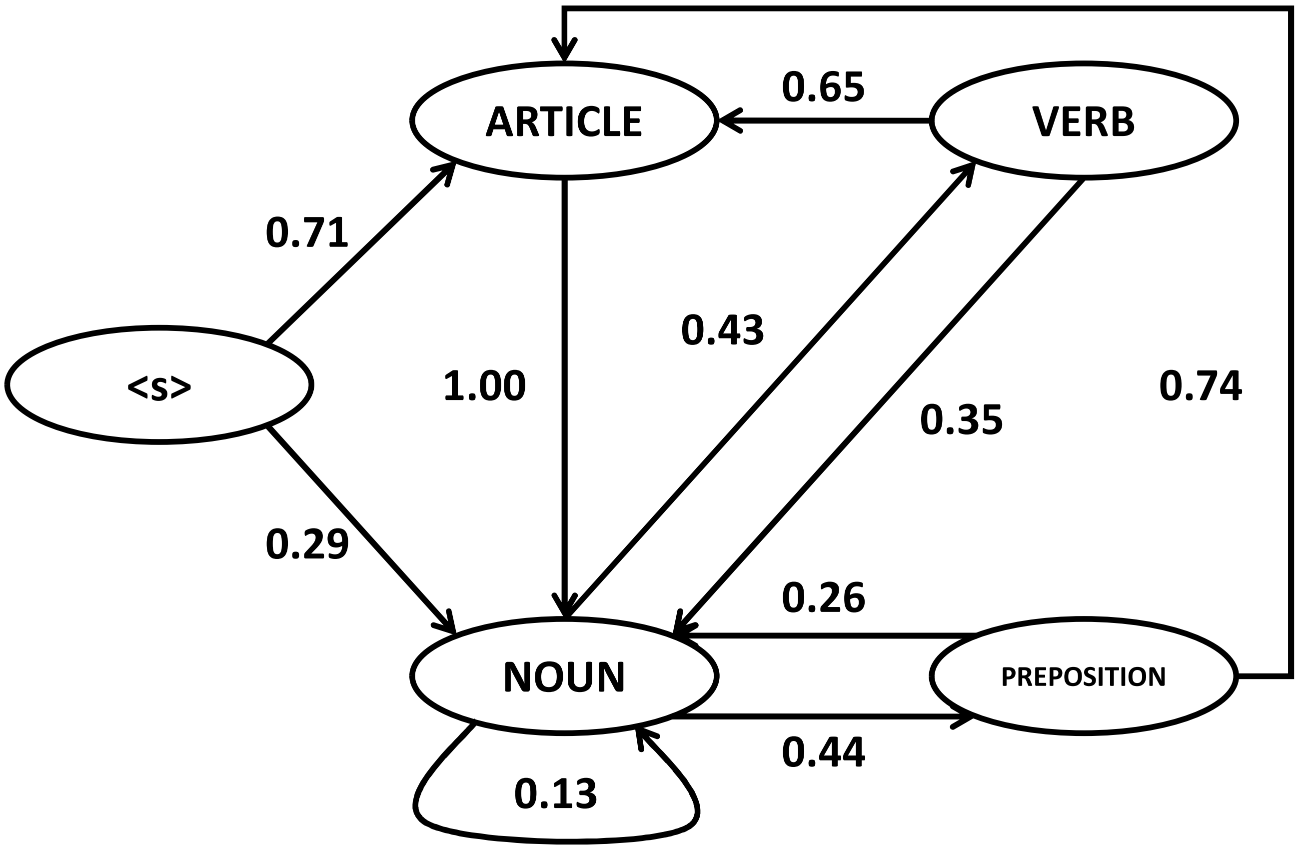


P(Bigram)	Estimate
P(ARTICLE|<S>)	0.71
P(NOUN|<S>)	0.29
P(NOUN|ARTICLE)	1.00
P(VERB|NOUN)	0.43
P(NOUN|NOUN)	0.13
P(PREPOSITION|NOUN)	0.44
P(NOUN|VERB)	0.35
P(ARTICLE|VERB)	0.65
P(ARTICLE|PREPOSITION)	0.74
P(NOUN|PREPOSITION)	0.26

For any sequence of categories (tags), their probability is:

$$P(C_1, C_2, C_3, \ldots, C_T) \cong \prod_{i=1}^{T} P(C_i \mid C_{i-1})$$

Hidden Markov Model

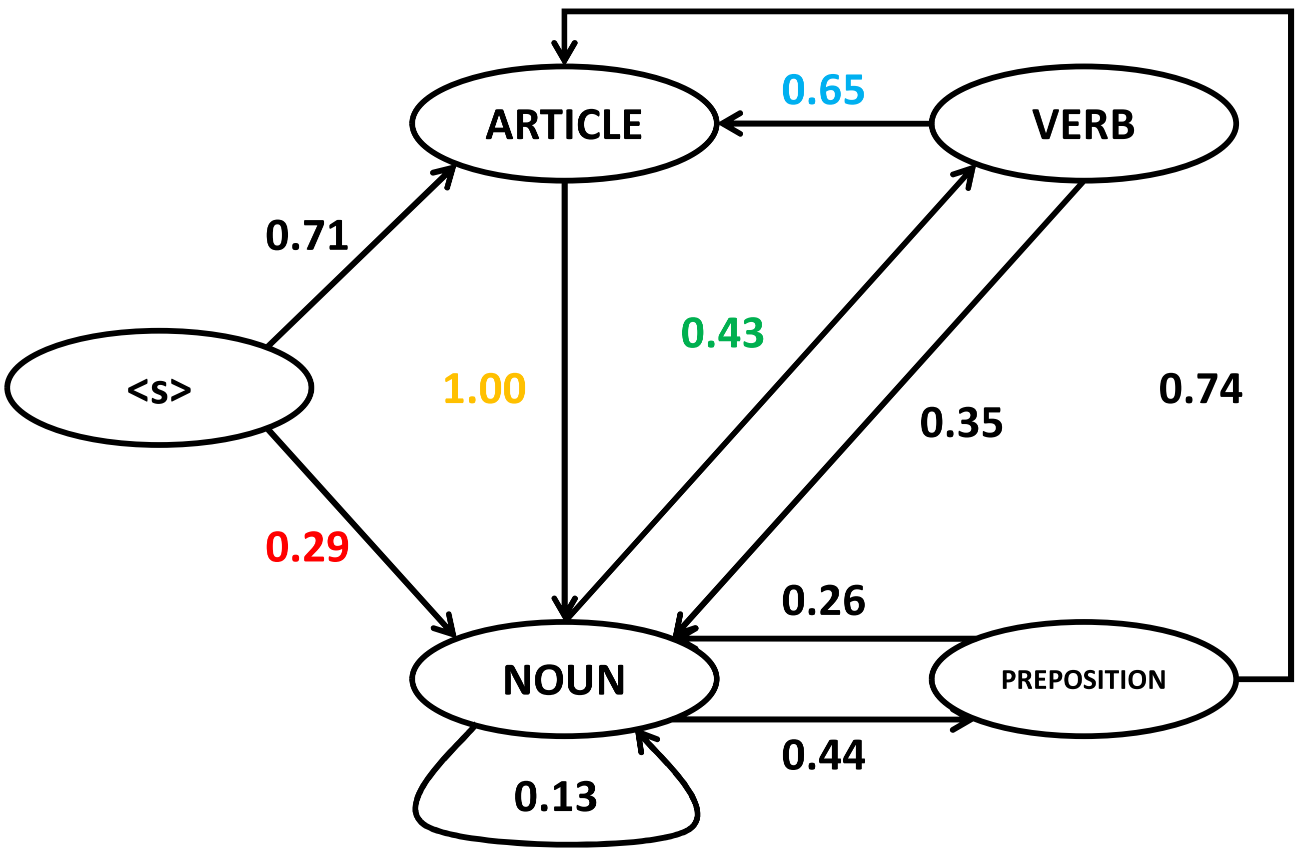


P(Bigram)	Estimate
P(ARTICLE|<S>)	0.71
P(NOUN|<S>)	0.29
P(NOUN|ARTICLE)	1.00
P(VERB|NOUN)	0.43
P(NOUN|NOUN)	0.13
P(PREPOSITION|NOUN)	0.44
P(NOUN|VERB)	0.35
P(ARTICLE|VERB)	0.65
P(ARTICLE|PREPOSITION)	0.74
P(NOUN|PREPOSITION)	0.26

For any sequence of categories (tags), their probability is:

$$\prod_{i=1}^{T} P(C_i \mid C_{i-1}) = P(N| < s >) * (V|N) * (ART|V) * (N|ART)$$

Hidden Markov Model



P(Bigram)	Estimate
P(ARTICLE|<S>)	0.71
P(NOUN|<S>)	0.29
P(NOUN|ARTICLE)	1.00
P(VERB|NOUN)	0.43
P(NOUN|NOUN)	0.13
P(PREPOSITION|NOUN)	0.44
P(NOUN|VERB)	0.35
P(ARTICLE|VERB)	0.65
P(ARTICLE|PREPOSITION)	0.74
P(NOUN|PREPOSITION)	0.26

For any sequence of categories (tags), their probability is:

$$\prod_{i=1}^{T} P(C_i \mid C_{i-1}) = 0.29 * 0.43 * 0.65 * 1.00 = 0.081$$

Example

Given our synthetic corpus, what is the most like sequence of categories (tags) corresponding to a sentence:

Flies like a flower

For example:

$$P(Flies, like, a, flower \mid N, V, ART, N) * P(N, V, ART, N)$$

$$\cong 5.4 * 10^{-5} * 0.081$$

POS Tagging: Simple Tagset

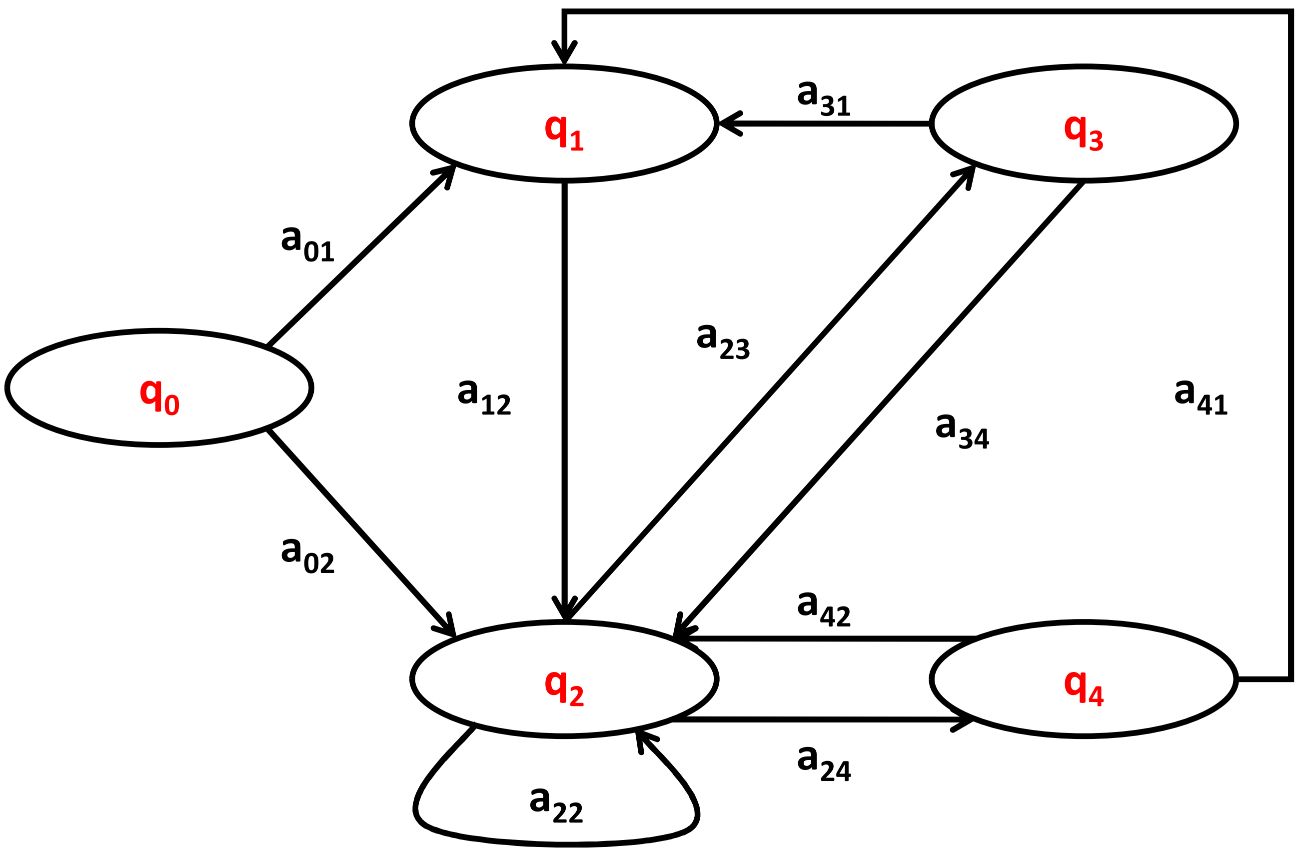
Let's assume we have a simple tagset:

- **N - NOUN**
- **V - VERB**
- **ART - ARTICLE**
- **P - PREPOSITION**

and some synthetic corpus. Ex. sentence:

Flies like a flower

Hidden Markov Model



Transition probability matrix A

	q_0	q_1	q_2	q_3	q_4	Notes
q_0	a_{00}	a_{01}	a_{02}	a_{03}	a_{04}	row sum = 1
q_1	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	row sum = 1
q_2	a_{20}	a_{21}	a_{22}	a_{23}	a_{24}	row sum = 1
q_3	a_{30}	a_{31}	a_{32}	a_{33}	a_{34}	row sum = 1
q_4	a_{40}	a_{41}	a_{42}	a_{43}	a_{44}	row sum = 1

HMMs are specified with:

- **A set of N states:**

  $Q = \{q_1, q_2, \ldots, q_N\}$

- **A transition probability matrix A, where each a_{ij} represents the probability of moving from state q_i to state q_j**
- **A sequence of observations O:**

  $O = o_1, o_2, \ldots, o_T$

- **A sequence of observation likelihoods (emission probabilities): probability of observation o_T being generated by a state q_i**

  $B = b_i(o_t)$

- **Special <s> and end (final) states**

  q_0 and q_F

POS Tagging: General Approach

Estimations with corpus counts:

$$P(C_i = VERB \mid C_{i-1} = NOUN) = \frac{Count\,(NOUN\ at\ position\ i-1\ and\ VERB\ at\ i)}{Count(NOUN\ at\ position\ i-1)}$$

Sample bigram probabilities from our synthetic corpus:

Category	Count at i	Pair	Count at i,i+1	P(Bigram)	Estimate
<s>	300	<s>, ARTICLE	213	P(ARTICLE|<S>)	0.71
<s>	300	<s>, NOUN	87	P(NOUN|<S>)	0.29
ARTICLE	558	ARTICLE, NOUN	558	P(NOUN|ARTICLE)	1.00
NOUN	833	NOUN, VERB	358	P(VERB|NOUN)	0.43
NOUN	833	[illegible]	108	P(NOUN|NOUN)	0.13
NOUN	833	NOUN, PREPOSITION	366	P(PREPOSITION|NOUN)	0.44
VERB	300	VERB, NOUN	75	P(NOUN|VERB)	0.35
VERB	300	VERB, ARTICLE	194	P(ARTICLE|VERB)	0.65
PREPOSITION	307	PREPOSITION, ARTICLE	226	P(ARTICLE|PREPOSITION)	0.74
PREPOSITION	307	PREPOSITION, NOUN	81	P(NOUN|PREPOSITION)	0.26

Transition probabilities

Synthetic Corpus: Word/Tag Counts

Summary of selected word counts in the synthetic corpus:

Word/Tag	N	V	ART	P	TOTAL
flies	21	23	0	0	44
fruit	49	5	1	0	55
like	10	30	0	21	61
a	1	0	201	0	202
the	1	0	300	2	303
flower	53	15	0	0	68
flowers	42	16	0	0	58
birds	64	1	0	0	65
others	592	210	56	284	1142
TOTAL	833	300	558	307	1998

From the table we can calculate lexical generation probabilities P(w|C) estimates:

P(*the*|ART) = 300/558 = 0.54

P(*flies*|N) = 21/833 = 0.025

P(*flies*|V) = 23/300 = 0.076

P(*like*|V) = 30/300 = 0.1

P(*like*|P) = 21/307 = 0.068

P(*a*|ART) = 201/558 = 0.36

P(*a*|N) = 1/833 = 0.001

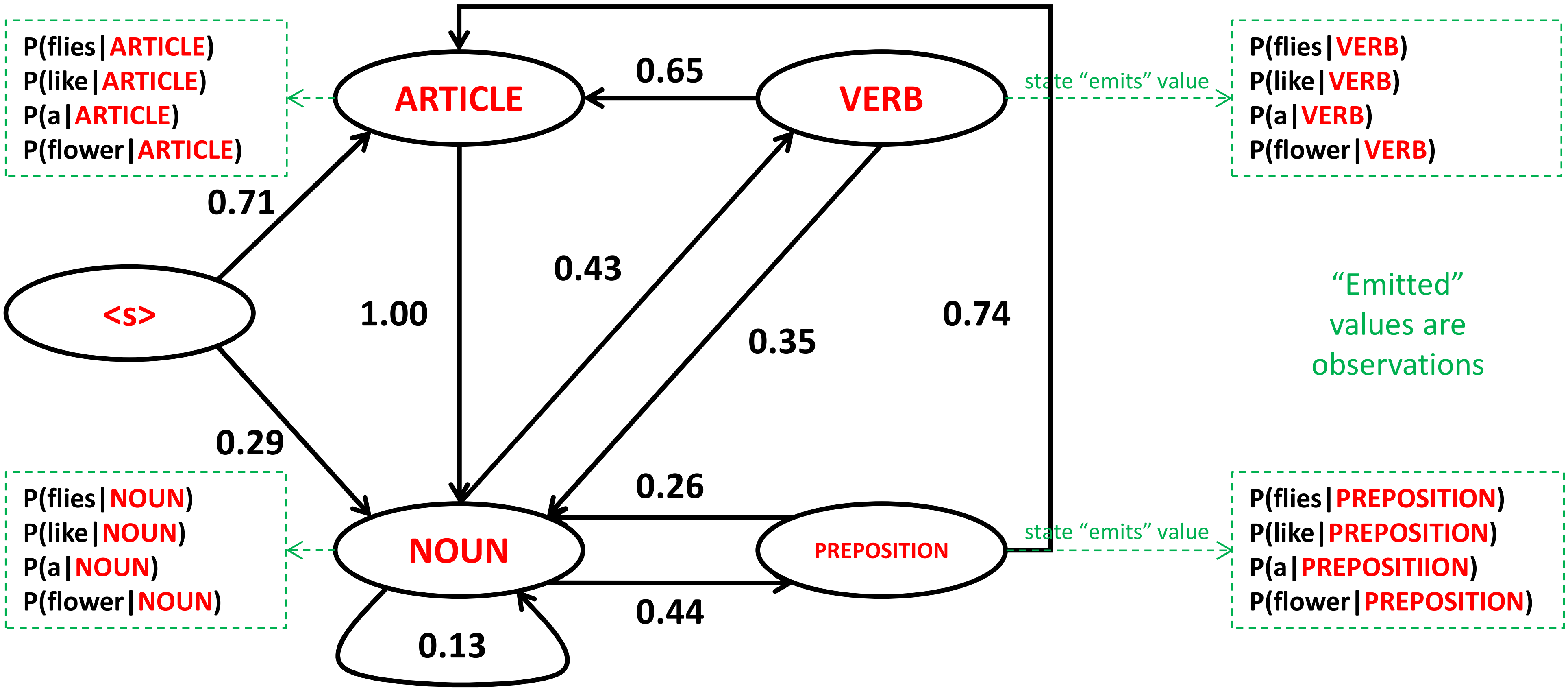
P(*flower*|N) = 53/833 = 0.063

P(*flower*|V) = 15/300 = 0.05

P(*like*|N) = 10/833 = 0.012

Emission probabilities

Hidden Markov Model



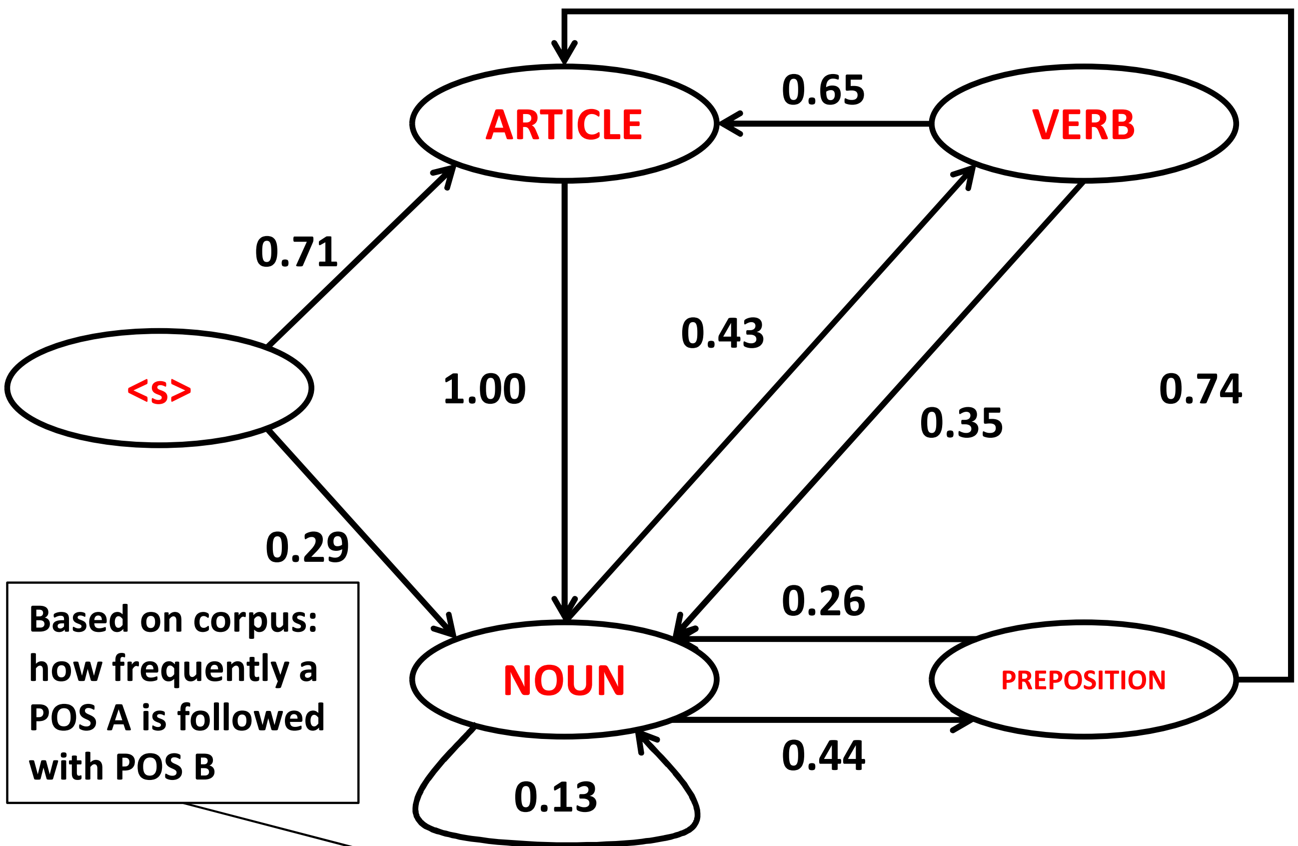
Transition probability matrix

	<s>	ARTICLE	NOUN	VERB	PREPOSITION
<s>	0.00	0.71	0.29	0.00	0.00
ARTICLE	0.00	0.00	1.00	0.00	0.00
NOUN	0.00	0.00	0.13	0.43	0.44
VERB	0.00	0.65	0.35	0.00	0.00
PREPOSITION	0.00	0.74	0.26	0.00	0.00

Emission probability matrix

	flies	like	a	flower
<s>	0.000	0.000	0.000	0.000
ARTICLE	0.000	0.000	0.360	0.000
NOUN	0.025	0.012	0.001	0.063
VERB	0.076	0.100	0.000	0.050
PREPOSITION	0.000	0.068	0.000	0.000

Hidden Markov Model



Based on corpus: how frequently a POS A is followed with POS B

Based on corpus: how frequently a word X is tagged with Y

Transition probability matrix	<s>	ARTICLE	NOUN	VERB	PREPOSITION
<s>	0.00	0.71	0.29	0.00	0.00
ARTICLE	0.00	0.00	1.00	0.00	0.00
NOUN	0.00	0.00	0.13	0.43	0.44
VERB	0.00	0.65	0.35	0.00	0.00
PREPOSITION	0.00	0.74	0.26	0.00	0.00

Emission probability matrix	flies	like	a	flower
<s>	0.000	0.000	0.000	0.000
ARTICLE	0.000	0.000	0.360	0.000
NOUN	0.025	0.012	0.001	0.063
VERB	0.076	0.100	0.000	0.050
PREPOSITION	0.000	0.068	0.000	0.000

Example

Given our synthetic corpus, what is the most like sequence of categories (tags) corresponding to a sentence:

Flies like a flower

We need to maximize:

$$P(w_1, w_2, w_3, \dots, w_T \mid C_1, C_2, C_3, \dots, C_T) * P(C_1, C_2, C_3, \dots, C_T) \cong$$

$$\cong \prod_{i=1}^{T} P(w_i \mid C_i) * P(C_i \mid C_{i-1})$$

Example

Given our synthetic corpus, what is the most like sequence of categories (tags) corresponding to a sentence:

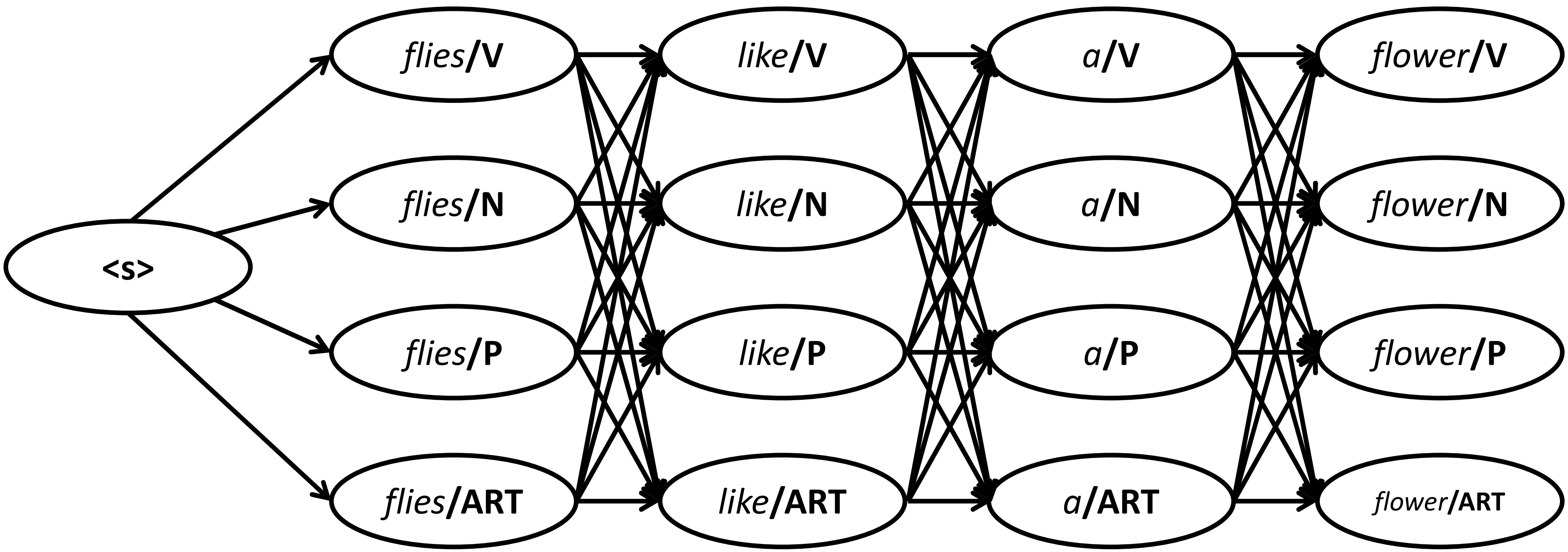
Flies like a flower

For example:

$$\boldsymbol{P(Flies, like, a, flower \mid N, V, ART, N) * P(N, V, ART, N)}$$

$$\boldsymbol{\cong 5.4 * 10^{-5} * 0.081}$$

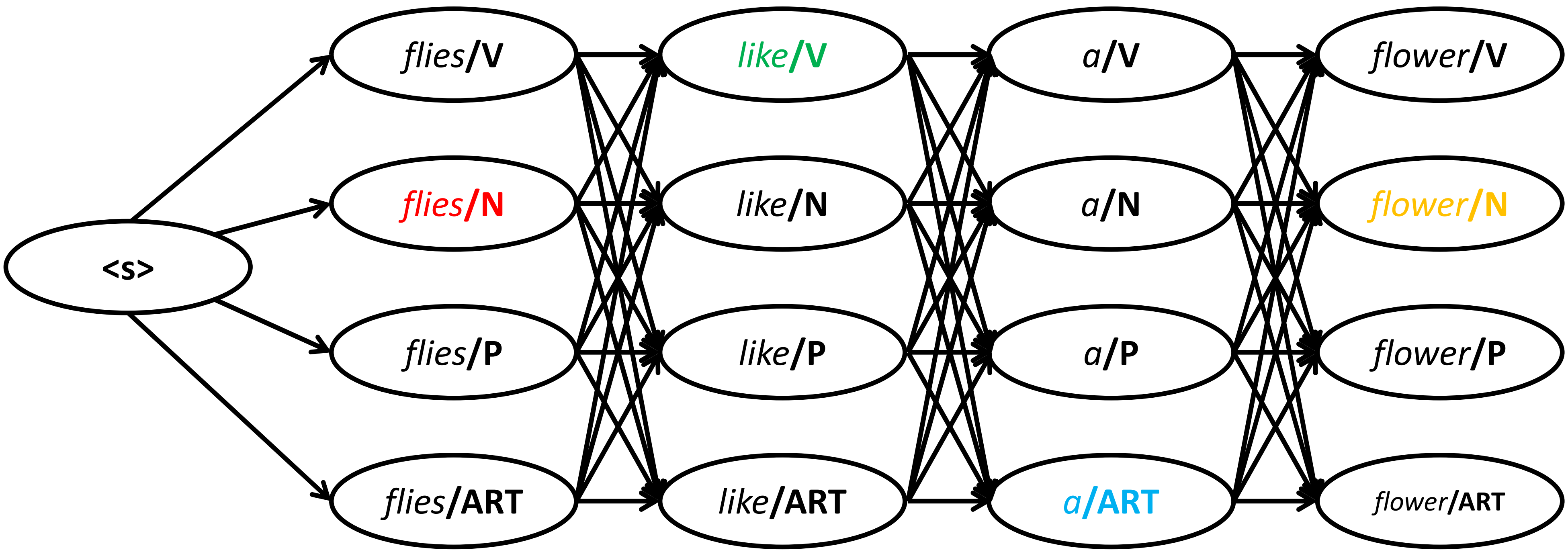
Example: All Possible Sequences



Every sequence can be assigned a probability:

$$P(w_1, w_2, w_3, \ldots, w_T \mid C_1, C_2, C_3, \ldots, C_T) \cong \prod_{i=1}^{T} P(w_i \mid C_i)$$

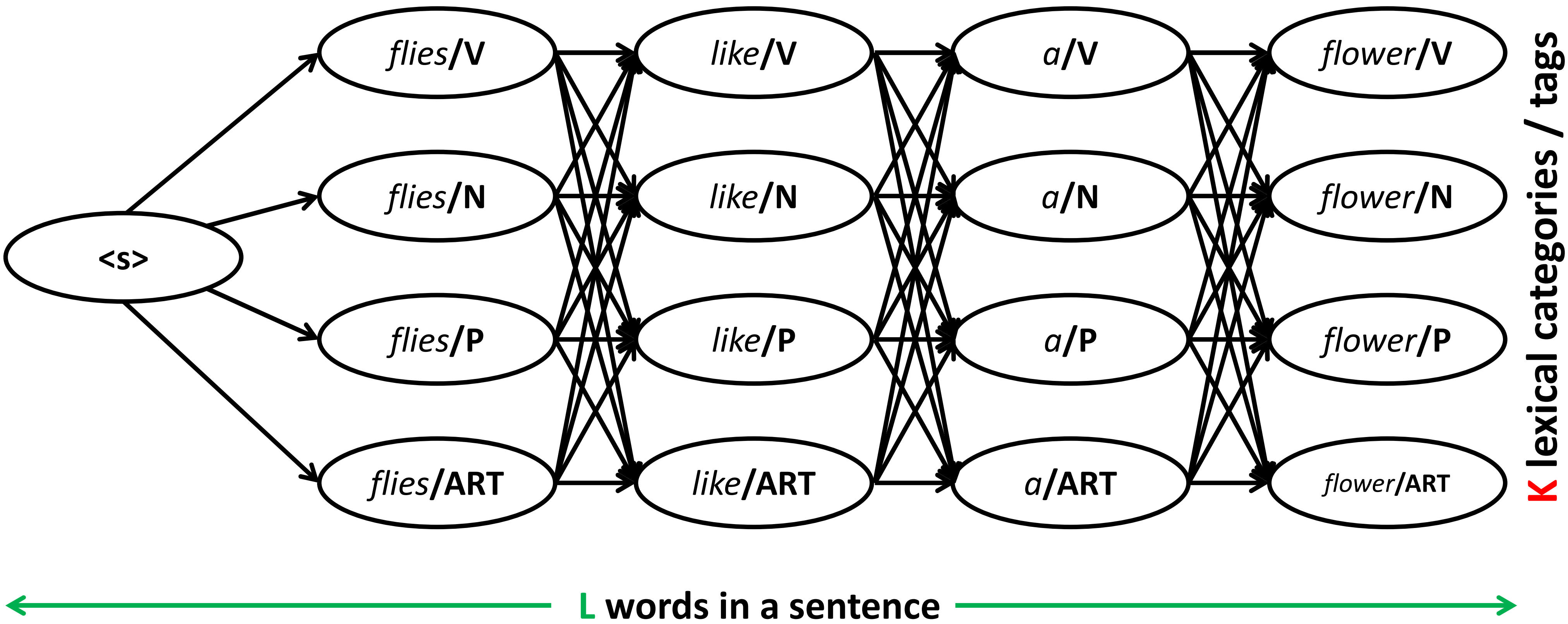
Example: Best Option



Best option will be:

$$\prod_{i=1}^{T} P(w_i \mid C_i) = P(flies|N) * P(like|V) * P(a|ART) * P(flower|N)$$

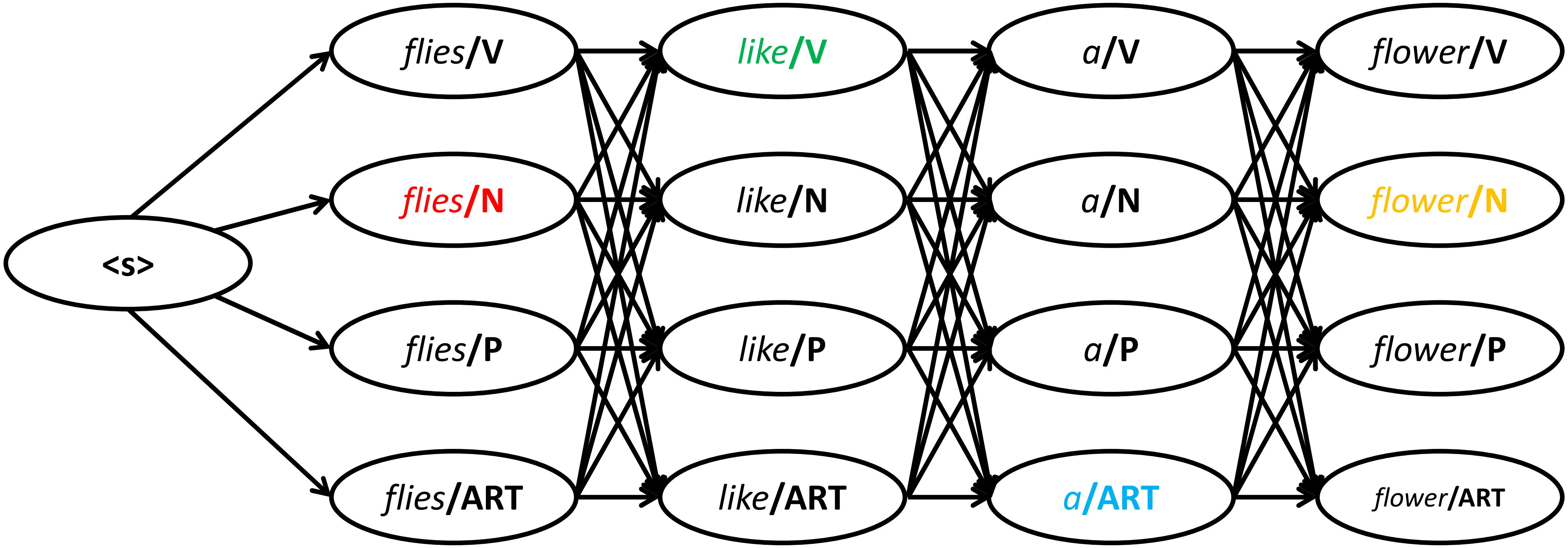
Example: All Possible Sequences



Brute force approach time complexity: $O(K^L)$

$K = 20$, $L = 10 \rightarrow 20^{10} = 10240000000000$

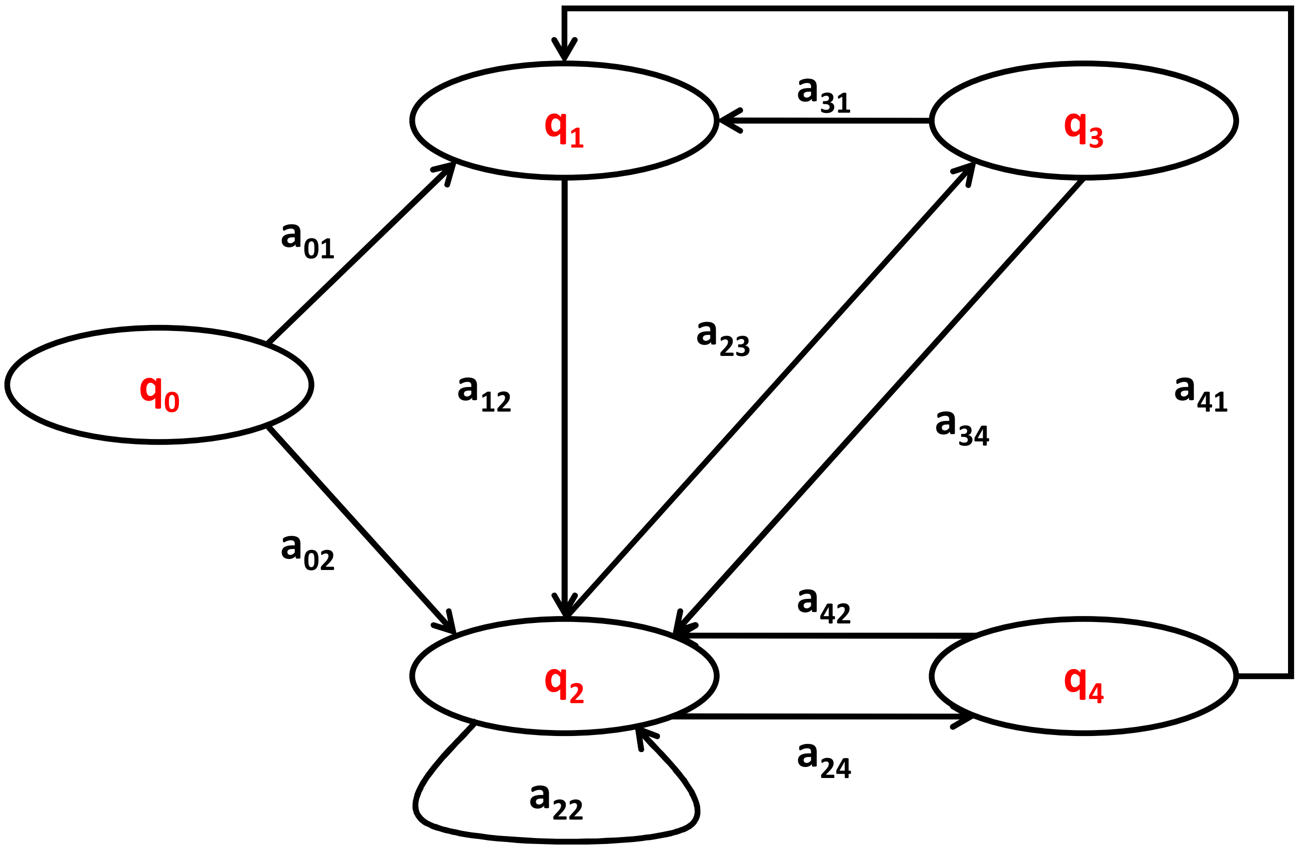
Example: Best Option



How can we efficiently find:

$$\prod_{i=1}^{T} P(w_i \mid C_i) = P(flies|N) * P(like|V) * P(a|ART) * P(flower|N)$$

Hidden Markov Model



Transition probability matrix A						
	q_0	q_1	q_2	q_3	q_4	Notes
q_0	a_{00}	a_{01}	a_{02}	a_{03}	a_{04}	row sum = 1
q_1	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	row sum = 1
q_2	a_{20}	a_{21}	a_{22}	a_{23}	a_{24}	row sum = 1
q_3	a_{30}	a_{31}	a_{32}	a_{33}	a_{34}	row sum = 1
q_4	a_{40}	a_{41}	a_{42}	a_{43}	a_{44}	row sum = 1

HMMs are specified with:

- **A set of N states:**

  $Q = \{q_1, q_2, \ldots, q_N\}$

- **A transition probability matrix A, where each a_{ij} represents the probability of moving from state q_i to state q_j**
- **A sequence of observations O:**

  $O = o_1, o_2, \ldots, o_T$

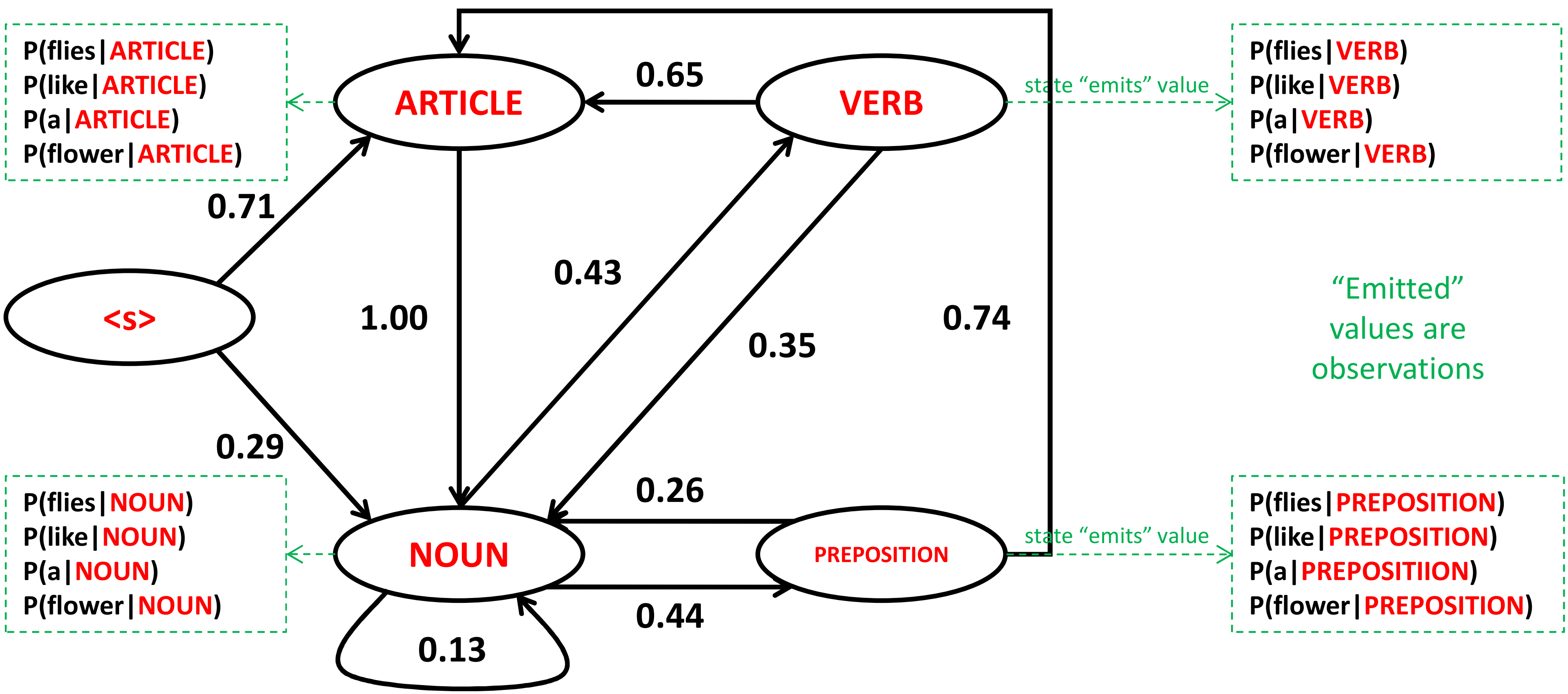
- **A sequence of observation likelihoods (emission probabilities): probability of observation o_T being generated by a state q_i**

  $B = b_i(o_t)$

- **Special <s> and end (final) states**

  q_0 and q_F

Hidden Markov Model



Transition probability matrix

	<s>	ARTICLE	NOUN	VERB	PREPOSITION
<s>	0.00	0.71	0.29	0.00	0.00
ARTICLE	0.00	0.00	1.00	0.00	0.00
NOUN	0.00	0.00	0.13	0.43	0.44
VERB	0.00	0.65	0.35	0.00	0.00
PREPOSITION	0.00	0.74	0.26	0.00	0.00

Emission probability matrix

	flies	like	a	flower
<s>	0.000	0.000	0.000	0.000
ARTICLE	0.000	0.000	0.360	0.000
NOUN	0.025	0.012	0.001	0.063
VERB	0.076	0.100	0.000	0.050
PREPOSITION	0.000	0.068	0.000	0.000

Hidden Markov Models: Decoding

The task of determining which sequence of variables is the underlying source of some sequence of observations is called the decoding:

Given as input an HMM $\alpha = (A, B)$ *and a sequence of observations* $o_1, o_2, \ldots, o_T$ *find the most probable sequence of states* $q_1, q_2, \ldots, q_T$.

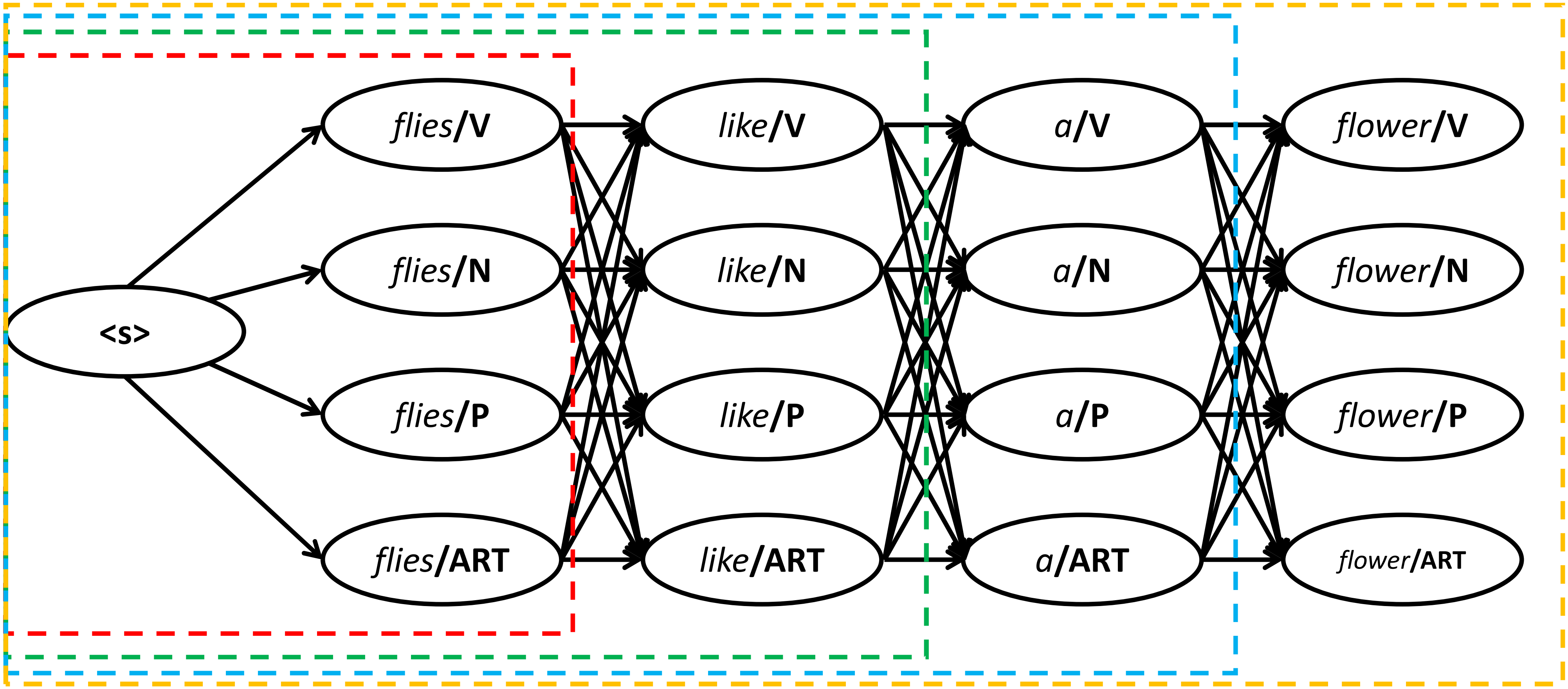
or in our case:

Given as input an HMM $\alpha = (A, B)$ *and a sequence of* ***words*** $w_1, w_2, \ldots, w_T$ *find the most probable sequence of* ***tags/states*** $C_1, C_2, \ldots, C_T$.

A - transition probabilities matrix

B - emission probabilities matrix

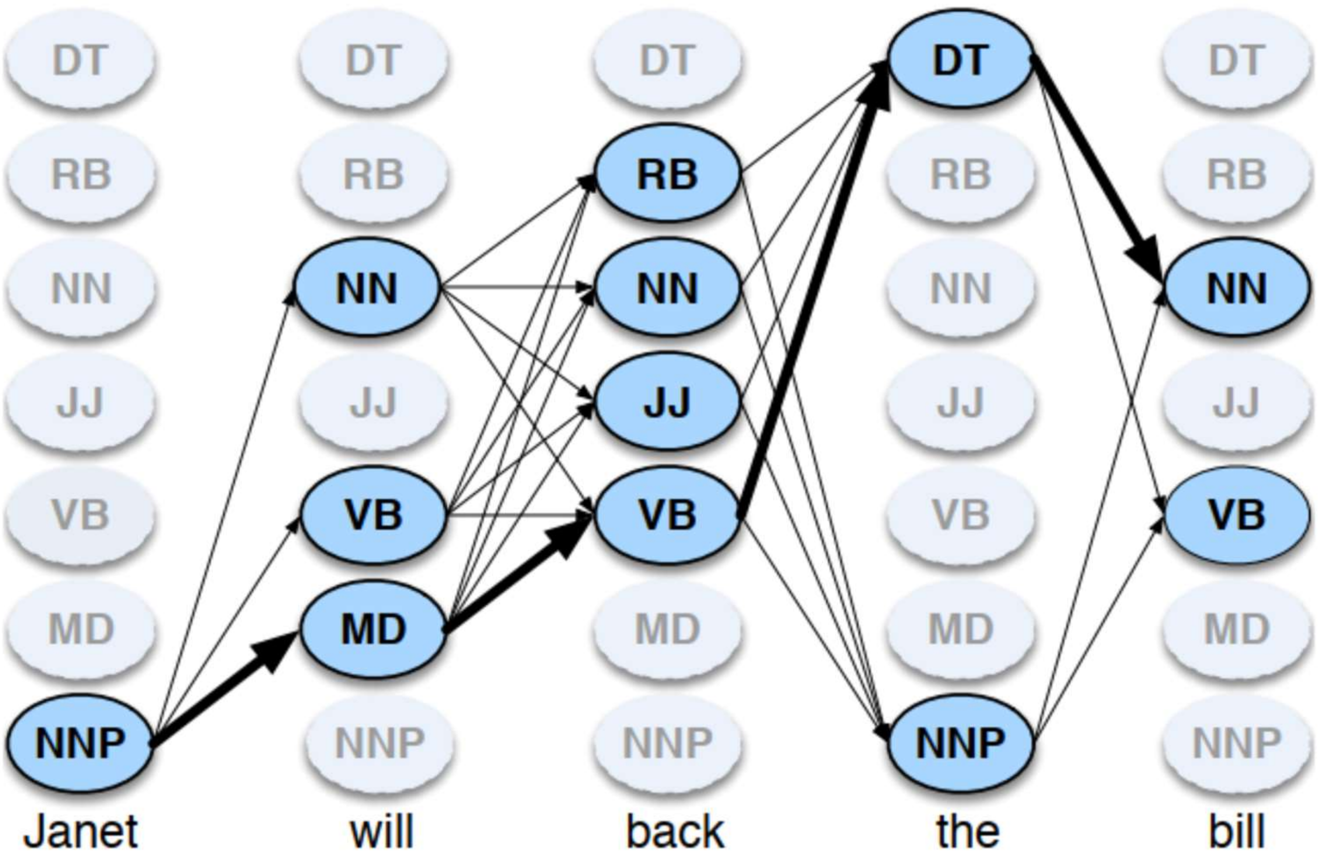
Viterbi Algorithm: the Idea



Maximizing [ ] **means maximizing** [ ], [ ], **and** [ ] .

In other words: maximize P() for all "sub-sentences".

Example: Possible Tag Sequences



Brown corpus tags:

NN - commn noun
NNP - singular proper noun
DT - singular determiner
VB - verb, base form
JJ - adjcective
MD - modal auxiliary
RB - adverb

	NNP	MD	VB	JJ	NN	RB	DT
<s>	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

Transition probabilities matrix

	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0	0
NN	0	0.000200	0.000223	0	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

Emission probabilities matrix

Viterbi Algorithm: Pseudocode

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

create a path probability matrix *viterbi[N,T]*
for each state s **from** 1 **to** N **do** ; initialization step
$\quad viterbi[s,1] \leftarrow \pi_s * b_s(o_1)$
$\quad backpointer[\mathrm{s},1] \leftarrow 0$
for each time step t **from** 2 **to** T **do** ; recursion step
  for each state s **from** 1 **to** N **do**
$$viterbi[\mathrm{s,t}] \leftarrow \max_{s'=1}^{N} \; viterbi[s',t-1] \; * \; a_{s',s} \; * \; b_s(o_t)$$
$$backpointer[\mathrm{s,t}] \leftarrow \operatorname*{argmax}_{s'=1}^{N} \; viterbi[s',t-1] \; * \; a_{s',s} \; * \; b_s(o_t)$$
$$bestpathprob \leftarrow \max_{s=1}^{N} \; viterbi[s,T]$$; termination step
$$bestpathpointer \leftarrow \operatorname*{argmax}_{s=1}^{N} \; viterbi[s,T]$$; termination step
bestpath $\leftarrow$ the path starting at state *bestpathpointer*, that follows backpointer[] to states back in time
return *bestpath*, *bestpathprob*

Viterbi Algorithm: Pseudocode

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

create a path probability matrix *viterbi[N,T]*
for each state s **from** 1 **to** N **do** ; initialization step
$\quad viterbi[s,1] \leftarrow \pi_s * b_s(o_1)$
$\quad backpointer[\mathrm{s},1] \leftarrow 0$
for each time step t **from** 2 **to** T **do** ; recursion step
$\quad$**for** each state s **from** 1 **to** N **do**

$$viterbi[\mathrm{s,t}] \leftarrow \max_{s'=1}^{N} \; viterbi[s', t-1] * a_{s',s} * b_s(o_t)$$

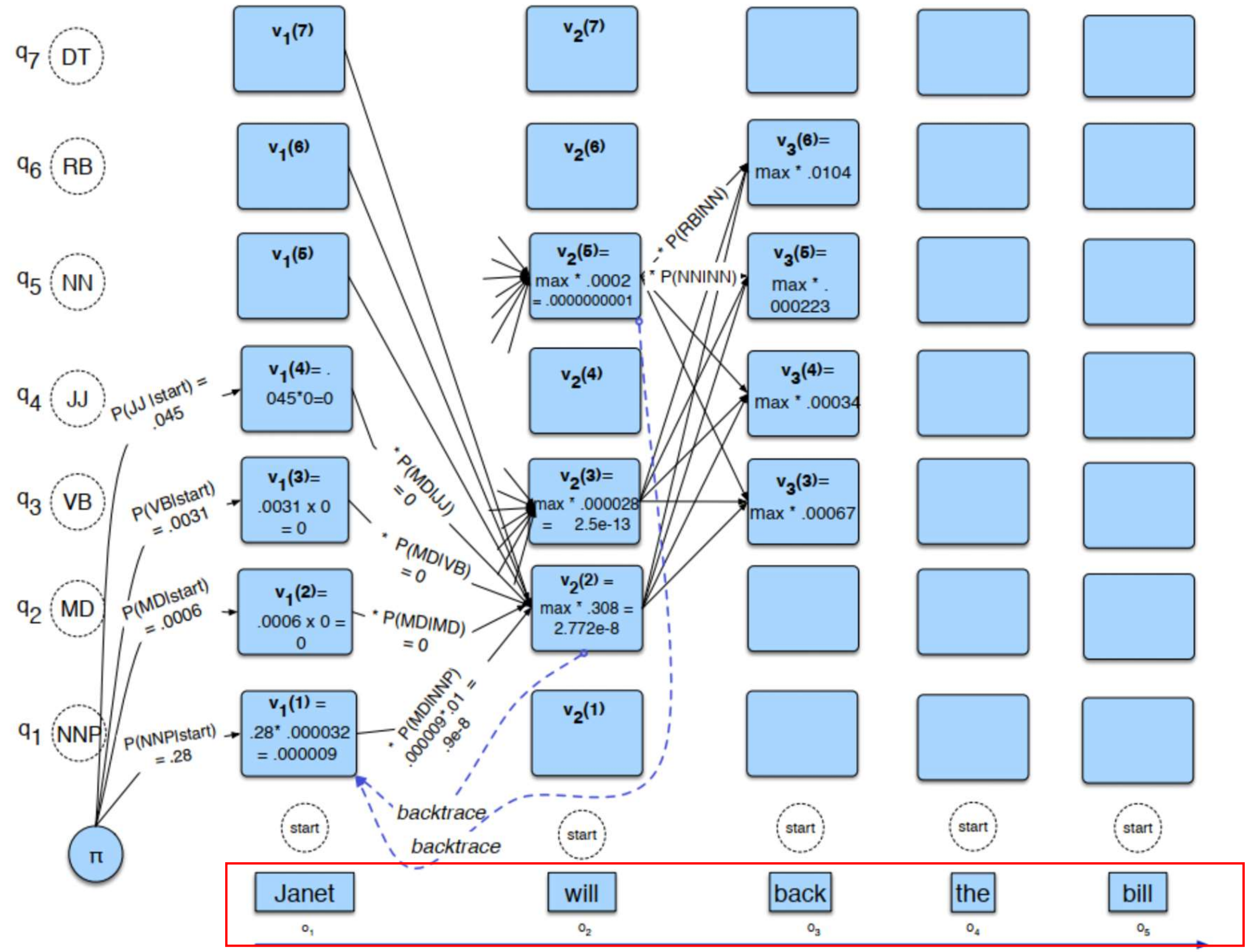
$$backpointer[\mathrm{s,t}] \leftarrow \operatorname{argmax}_{s'=1}^{N} \; viterbi[s', t-1] * a_{s',s} * b_s(o_t)$$

$$bestpathprob \leftarrow \max_{s=1}^{N} \; viterbi[s,T]$$; termination step

$$bestpathpointer \leftarrow \operatorname{argmax}_{s=1}^{N} \; viterbi[s,T]$$; termination step

bestpath $\leftarrow$ the path starting at state *bestpathpointer*, that follows backpointer[] to states back in time
return *bestpath*, *bestpathprob*

Viterbi Algorithm: Example



Viterbi Algorithm: Pseudocode

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

create a path probability matrix *viterbi[N,T]*

for each state s **from** 1 **to** N **do** ; initialization step

$\quad viterbi[s,1] \leftarrow \pi_s * b_s(o_1)$

$\quad backpointer[s,1] \leftarrow 0$

for each time step t **from** 2 **to** T **do** ; recursion step

for each state s **from** 1 **to** N **do**

$$viterbi[s,t] \leftarrow \max_{s'=1}^{N} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$$

$$backpointer[s,t] \leftarrow \underset{s'=1}{\overset{N}{\operatorname{argmax}}} \; viterbi[s',t-1] * a_{s',s} * b_s(o_t)$$

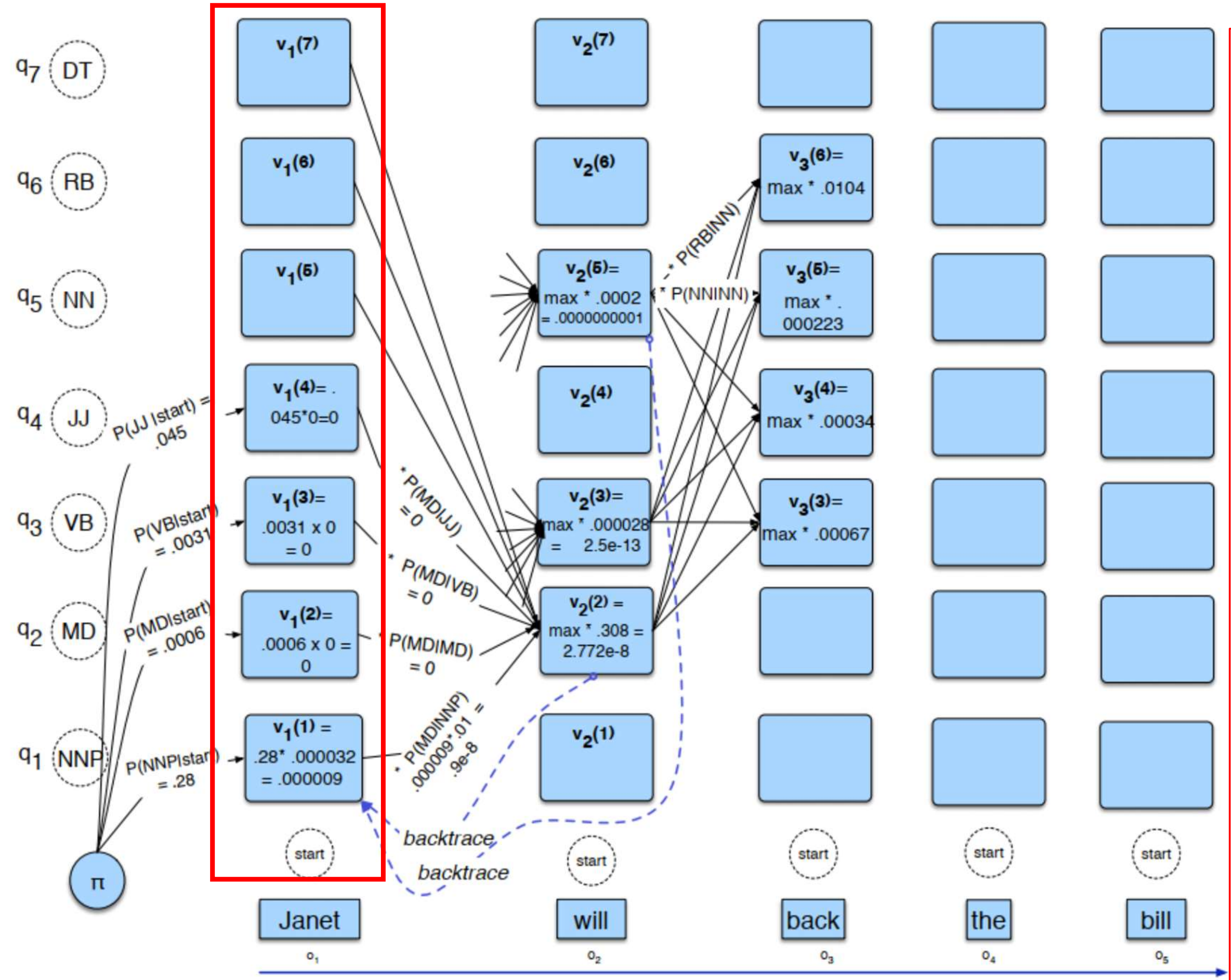
$bestpathprob \leftarrow \max_{s=1}^{N} viterbi[s,T]$; termination step

$bestpathpointer \leftarrow \underset{s=1}{\overset{N}{\operatorname{argmax}}} \; viterbi[s,T]$; termination step

bestpath $\leftarrow$ the path starting at state *bestpathpointer*, that follows backpointer[] to states back in time

return *bestpath*, *bestpathprob*

Viterbi Algorithm: Example



Initialization step:

$v_1(1) = v_1(\text{NNP}) = P(T)*P(E) = (\text{NNP}|\text{<s>})*P(\text{Janet}|\text{NNP})$

$v_1(2) = v_1(\text{MD}) = P(T)*P(E) = (\text{MD}|\text{<s>})*P(\text{Janet}|\text{MD})$

$v_1(3) = v_1(\text{VB}) = P(T)*P(E) = (\text{VB}|\text{<s>})*P(\text{Janet}|\text{VB})$

$v_1(4) = v_1(\text{JJ}) = P(T)*P(E) = (\text{JJ}|\text{<s>})*P(\text{Janet}|\text{JJ})$

$v_1(5) = v_1(\text{NN}) = P(T)*P(E) = (\text{NN}|\text{<s>})*P(\text{Janet}|\text{NN})$

$v_1(6) = v_1(\text{RB}) = P(T)*P(E) = (\text{RB}|\text{<s>})*P(\text{Janet}|\text{RB})$

$v_1(7) = v_1(\text{DT}) = P(T)*P(E) = (\text{DT}|\text{<s>})*P(\text{Janet}|\text{DT})$

P(T) - P(transition)
P(E) - P(emission)

Viterbi Algorithm: Example

Transition probabilities matrix

	NNP	MD	VB	JJ	NN	RB	DT
<s>	0.2767	0.0006	0.0031	0.0453	0.0449	0.0510	0.2026
NNP	0.3777	0.0110	0.0009	0.0084	0.0584	0.0090	0.0025
MD	0.0008	0.0002	0.7968	0.0005	0.0008	0.1698	0.0041
VB	0.0322	0.0005	0.0050	0.0837	0.0615	0.0514	0.2231
JJ	0.0366	0.0004	0.0001	0.0733	0.4509	0.0036	0.0036
NN	0.0096	0.0176	0.0014	0.0086	0.1216	0.0177	0.0068
RB	0.0068	0.0102	0.1011	0.1012	0.0120	0.0728	0.0479
DT	0.1147	0.0021	0.0002	0.2157	0.4744	0.0102	0.0017

Emission probabilities matrix

	Janet	will	back	the	bill
NNP	0.000032	0	0	0.000048	0
MD	0	0.308431	0	0	0
VB	0	0.000028	0.000672	0	0.000028
JJ	0	0	0.000340	0	0
NN	0	0.000200	0.000223	0	0.002337
RB	0	0	0.010446	0	0
DT	0	0	0	0.506099	0

Initialization step:

$v_1(1) = v_1(NNP) = P(T)*P(E) = (NNP|<s>)*P(Janet|NNP)$

$v_1(2) = v_1(MD) = P(T)*P(E) = (MD|<s>)*P(Janet|MD)$

$v_1(3) = v_1(VB) = P(T)*P(E) = (VB|<s>)*P(Janet|VB)$

$v_1(4) = v_1(JJ) = P(T)*P(E) = (JJ|<s>)*P(Janet|JJ)$

$v_1(5) = v_1(NN) = P(T)*P(E) = (NN|<s>)*P(Janet|NN)$

$v_1(6) = v_1(RB) = P(T)*P(E) = (RB|<s>)*P(Janet|RB)$

$v_1(7) = v_1(DT) = P(T)*P(E) = (DT|<s>)*P(Janet|DT)$

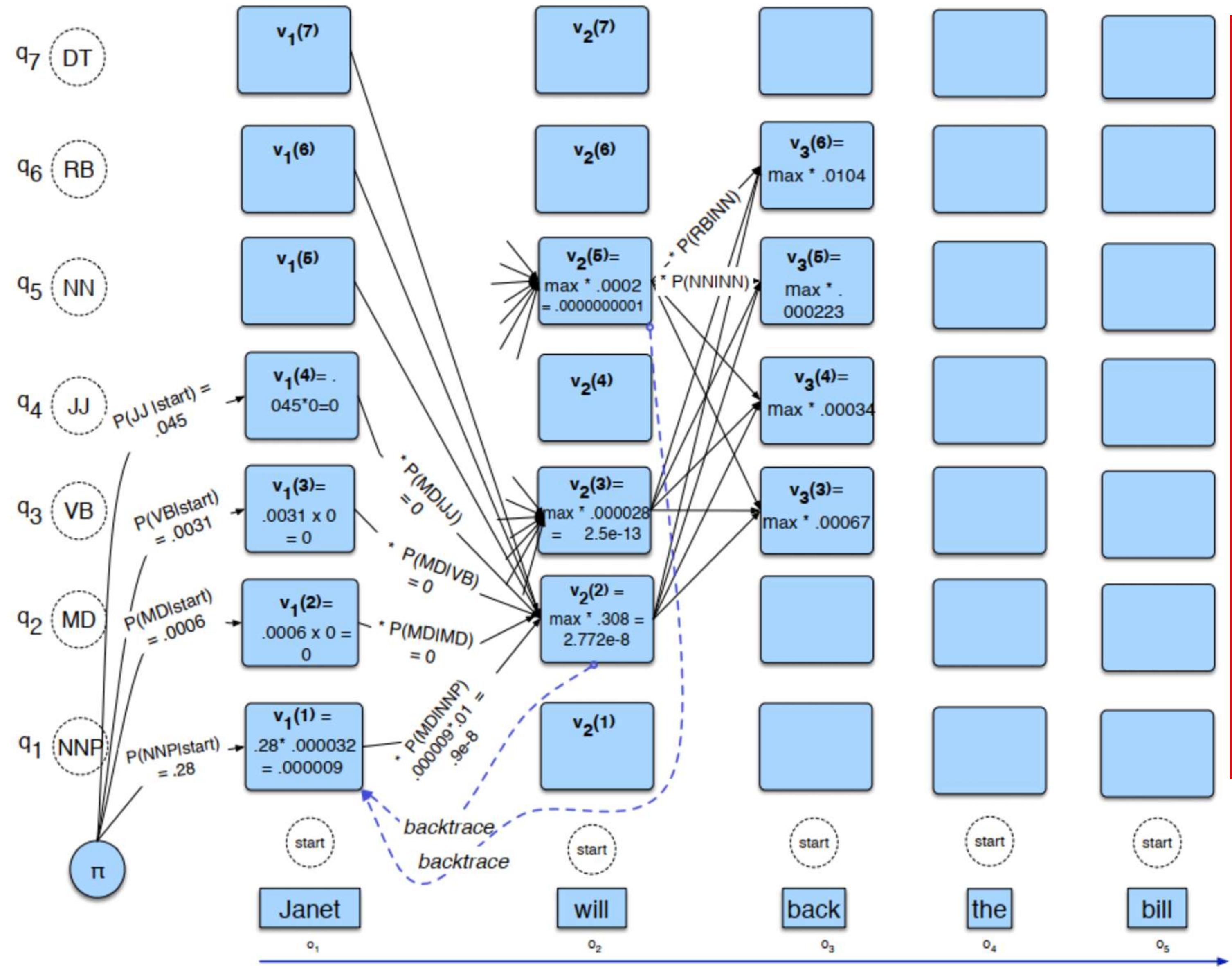
P(T) - P(transition)
P(E) - P(emission)

Viterbi Algorithm: Pseudocode

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

create a path probability matrix *viterbi[N,T]*
for each state s **from** 1 **to** N **do** ; initialization step
    $viterbi[s,1] \leftarrow \pi_s \, * \, b_s(o_1)$
    $backpointer[s,1] \leftarrow 0$
for each time step t **from** 2 **to** T **do** ; recursion step
  for each state s **from** 1 **to** N **do**
    $$viterbi[s,t] \leftarrow \max_{s'=1}^{N} \; viterbi[s',t-1] \, * \, a_{s',s} \, * \, b_s(o_t)$$
    $$backpointer[s,t] \leftarrow \operatorname*{argmax}_{s'=1}^{N} \; viterbi[s',t-1] \, * \, a_{s',s} \, * \, b_s(o_t)$$
$$bestpathprob \leftarrow \max_{s=1}^{N} \; viterbi[s,T]$$; termination step
$$bestpathpointer \leftarrow \operatorname*{argmax}_{s=1}^{N} \; viterbi[s,T]$$; termination step
bestpath ← the path starting at state *bestpathpointer*, that follows backpointer[] to states back in time
return *bestpath*, *bestpathprob*

Viterbi Algorithm: Example



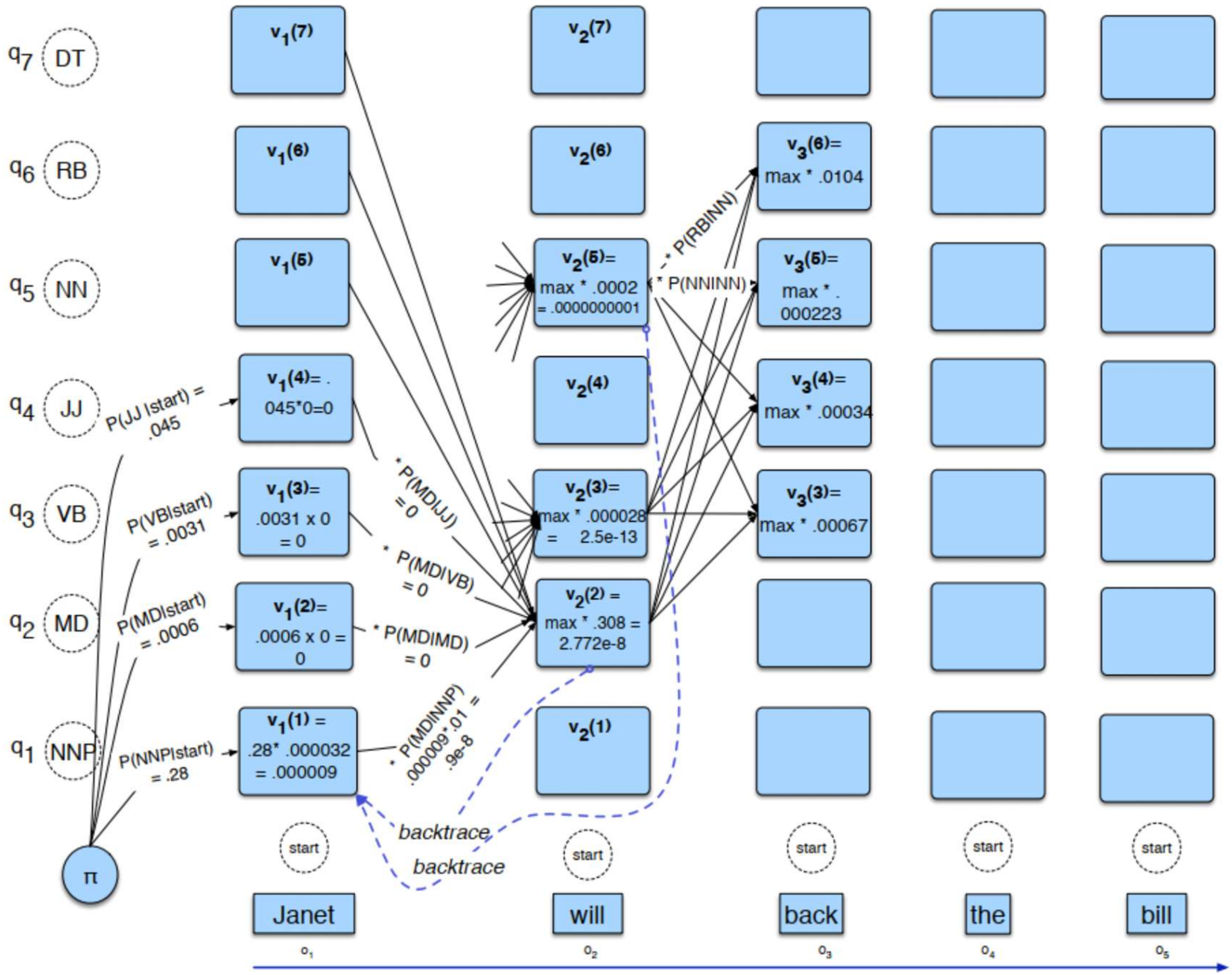
$v_{t-1}(j)$ - previous Viterbi path probability (from the previous time step)

a_{ij} - transition probability from previous state q_i to state q_j

$b_j(o_t)$ - observation likelihood of the observation symbol o_t given the current state q_j

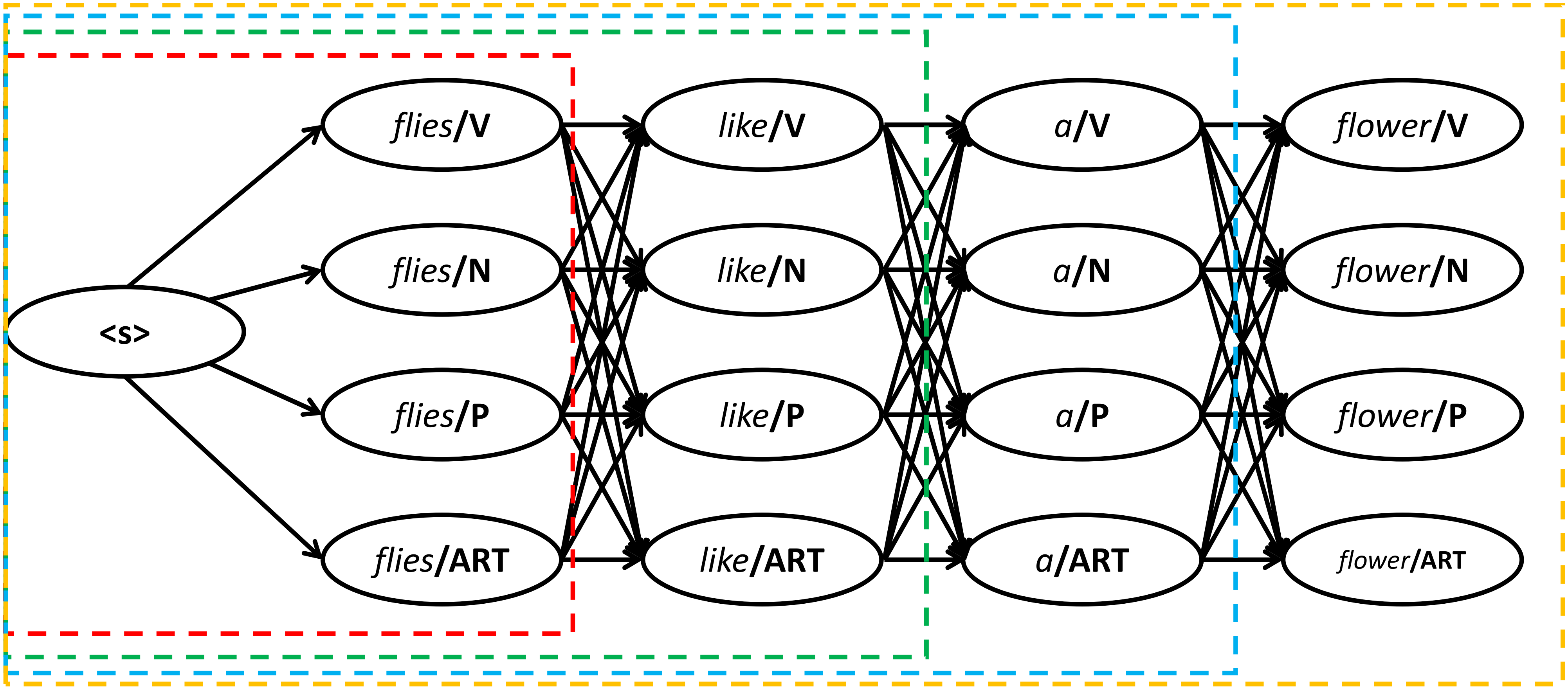
$$For\ each\ state\ q_j\ at\ time\ t, compute{:}\ v_t(j) = \max_{i=1toN} v_{t-1}(i) * a_{ij} * b_j(o_t)$$

Viterbi Algorithm: Example



For each state q_j *at time* t*, compute:* $v_t(j) = \max_{i=1t} v_{t-1}(i) * P(transition) * P(emisssion)$

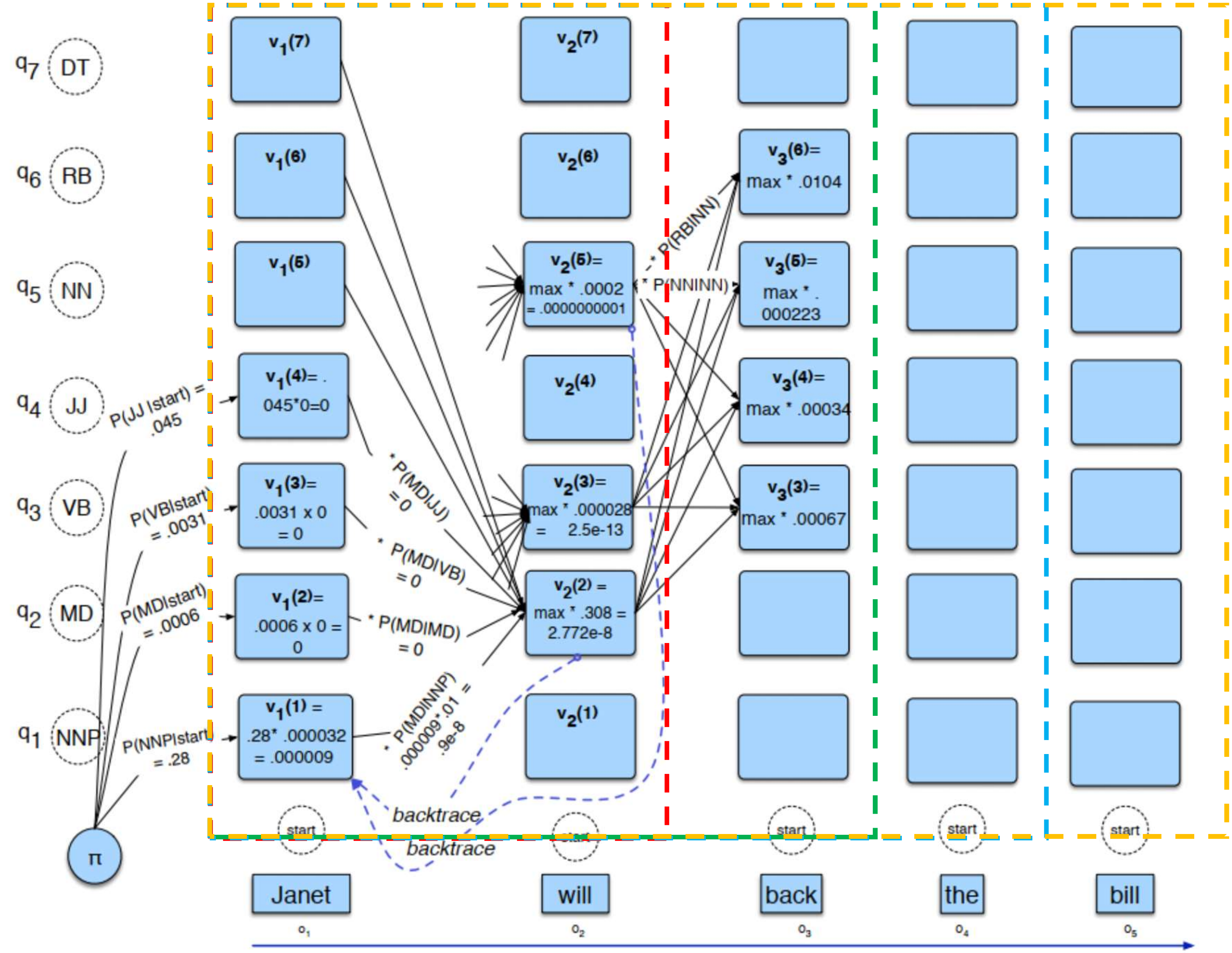
Viterbi Algorithm: the Idea



Maximizing [ ] means maximizing [ ], [ ], and [ ].

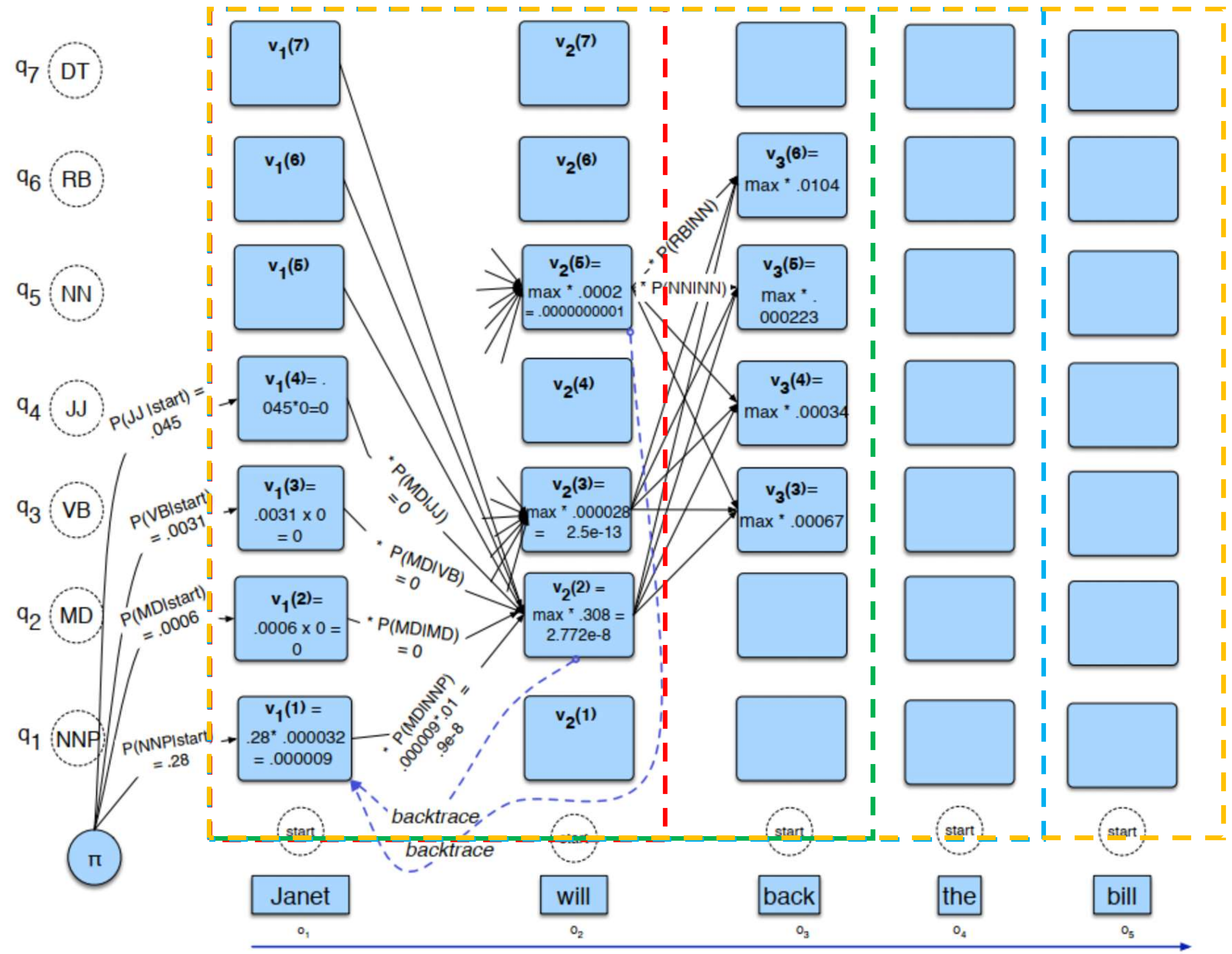
In other words: maximize P() for all "sub-sentences".

Viterbi Algorithm: Example



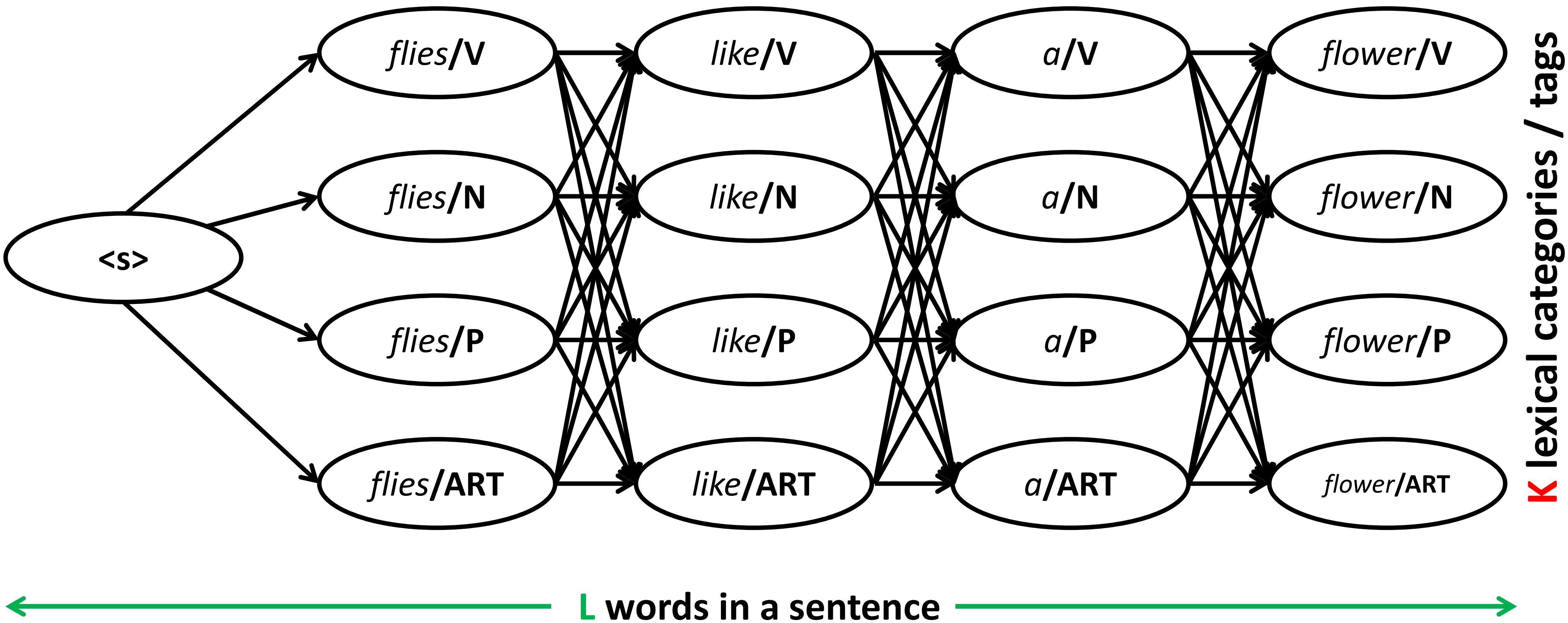
Maximizing [yellow box] **means maximizing** [blue box] **,** [green box] **, and** [red box] **.**

Viterbi Algorithm: Example



Some paths **need not to be explored**!

Example: Viterbi



Viterbi algorithm approach time complexity: O(L * K²)

K = 20, L = 10 → 10 * 20² = 4000

versus brute force approach time complexity: O(K^L)

K = 20, L = 10 → 20¹⁰ = 10240000000000