THE UNIVERSITY OF MELBOURNE

Comp90042 Workshop Week 8

9 May


1929

Table of Contents

Probabilistic context-free grammar

What is a **probabilistic context-free grammar** and what problem does it attempt to solve?

- A **probabilistic context-free grammar** adds probability to each production rule in the context-free grammar.

- Provides a "language model" to describe the likely sentences in a language.

CYK Parsing

an	*park*	*by*	*Bob*	*walked*	*an*	*park*	*with*	*Bob*
[0,1] Det	[0,2] NP	[0,3] -	[0,4] NP, X	[0,5] -	[0,6] -	[0,7] S	[0,8] -	[0,9] S, S
	[1,2] N	[1,3] -	[1,4] Y	[1,5] -	[1,6] -	[1,7] -	[1,8] -	[1,9] -
		[2,3] P	[2,4] PP	[2,5] -	[2,6] -	[2,7] -	[2,8] -	[2,9] -
			[3,4] NP	[3,5] -	[3,6] -	[3,7] S	[3,8] -	[3,9] S, S
				[4,5] V	[4,6] -	[4,7] VP	[4,8] -	[4,9] VP, VP
					[5,6] Det	[5,7] NP	[5,8] -	[5,9] NP, X
						[6,7] N	[6,8] -	[6,9] Y
							[7,8] P	[7,9] PP
								[8,9] NP

- How many valid parse trees?
- Which one is better? How can we show the difference?

Probabilistic CYK parsing vs CYK parsing

Probabilistic CYK parsing: In addition to CYK, we add probabilities to production rules.

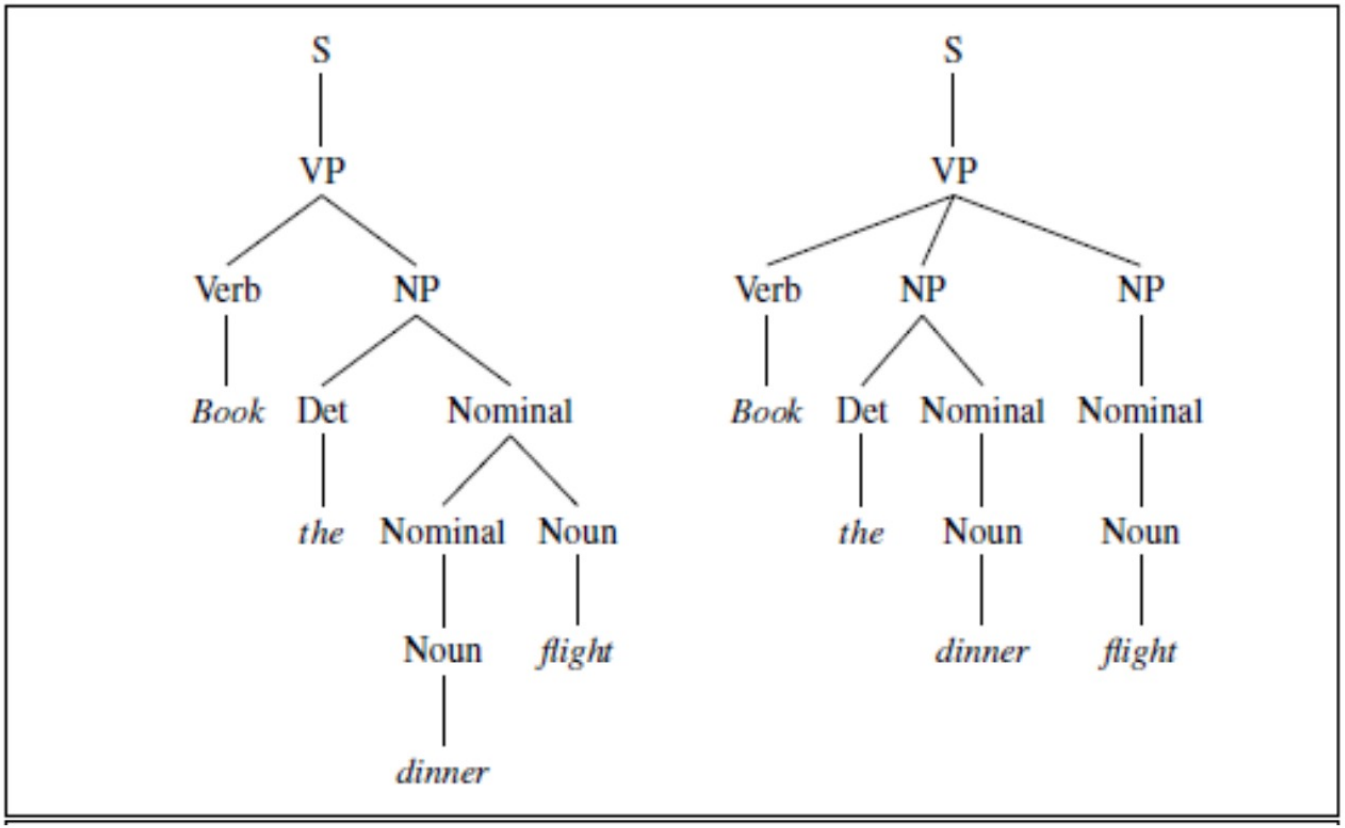
Book the dinner flight

A. Book [the dinner flight] – book the flight that serves dinner

B. Book [the dinner] [flight] – book whom? [the dinner] what? [flight]

Probabilistic CYK parsing vs CYK parsing

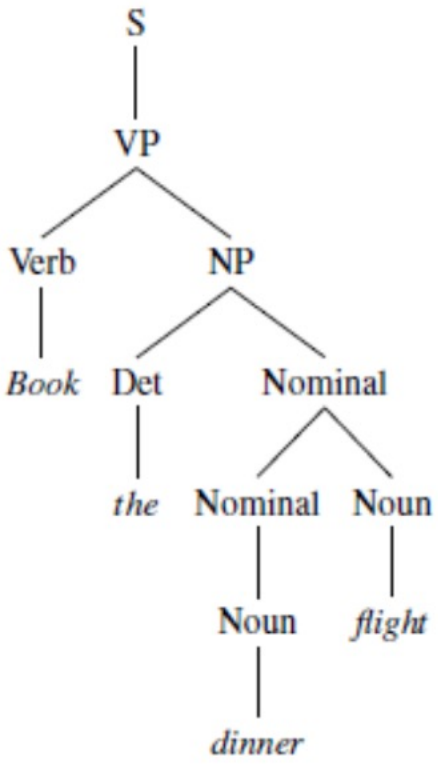
Book the dinner flight



Probabilistic CYK parsing vs CYK parsing

Book the dinner flight : VP -> Verb NP or Verb NP NP?

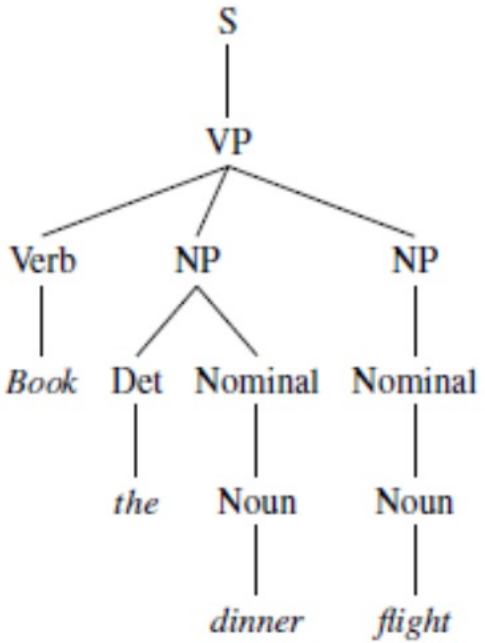
Grammar		Lexicon
$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow$ *that* [.10] | *a* [.30] | *the* [.60]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow$ *book* [.10] | *flight* [.30]
$S \rightarrow VP$	[.05]	| *meal* [.05] | *money* [.05]
$NP \rightarrow Pronoun$	[.35]	| *flight* [.40] | *dinner* [.10]
$NP \rightarrow Proper\text{-}Noun$	[.30]	$Verb \rightarrow$ *book* [.30] | *include* [.30]
$NP \rightarrow Det\ Nominal$	[.20]	| *prefer* [.40]
$NP \rightarrow Nominal$	[.15]	$Pronoun \rightarrow$ *I* [.40] | *she* [.05]
$Nominal \rightarrow Noun$	[.75]	| *me* [.15] | *you* [.40]
$Nominal \rightarrow Nominal\ Noun$	[.20]	$Proper\text{-}Noun \rightarrow$ *Houston* [.60]
$Nominal \rightarrow Nominal\ PP$	[.05]	| *NWA* [.40]
$VP \rightarrow Verb$	[.35]	$Aux \rightarrow$ *does* [.60] | *can* [.40]
$VP \rightarrow Verb\ NP$	[.20]	$Preposition \rightarrow$ *from* [.30] | *to* [.30]
$VP \rightarrow Verb\ NP\ PP$	[.10]	| *on* [.20] | *near* [.15]
$VP \rightarrow Verb\ PP$	[.15]	| *through* [.05]
$VP \rightarrow Verb\ NP\ NP$	[.05]	
$VP \rightarrow VP\ PP$	[.15]	
$PP \rightarrow Preposition\ NP$	[1.0]	



Probabilistic CYK parsing vs CYK parsing

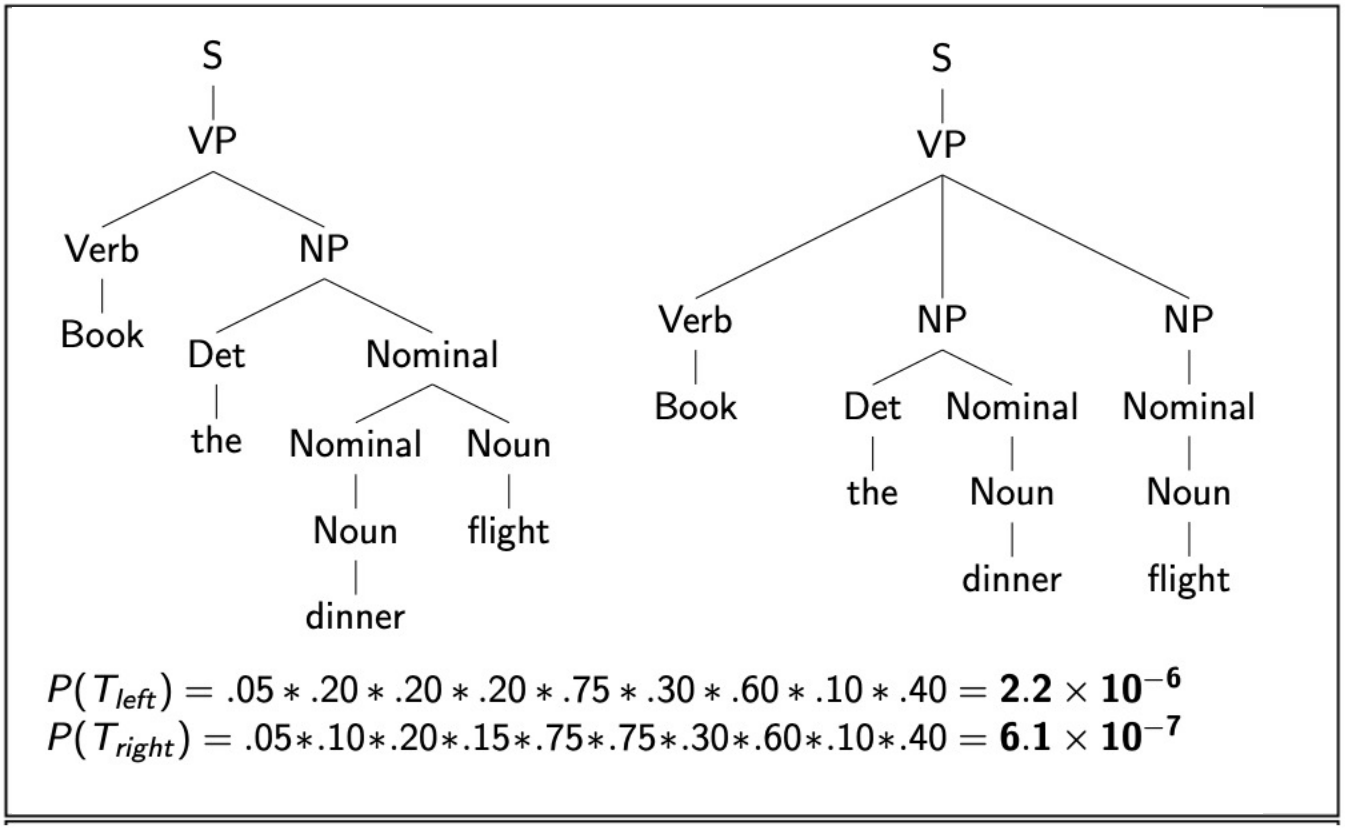
Book the dinner flight : VP -> Verb NP or Verb NP NP?

Grammar		Lexicon
$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that$ [.10] | a [.30] | the [.60]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book$ [.10] | $flight$ [.30]
$S \rightarrow VP$	[.05]	| $meal$ [.05] | $money$ [.05]
$NP \rightarrow Pronoun$	[.35]	| $flight$ [.40] | $dinner$ [.10]
$NP \rightarrow Proper\text{-}Noun$	[.30]	$Verb \rightarrow book$ [.30] | $include$ [.30]
$NP \rightarrow Det\ Nominal$	[.20]	| $prefer$ [.40]
$NP \rightarrow Nominal$	[.15]	$Pronoun \rightarrow I$ [.40] | she [.05]
$Nominal \rightarrow Noun$	[.75]	| me [.15] | you [.40]
$Nominal \rightarrow Nominal\ Noun$	[.20]	$Proper\text{-}Noun \rightarrow Houston$ [.60]
$Nominal \rightarrow Nominal\ PP$	[.05]	| NWA [.40]
$VP \rightarrow Verb$	[.35]	$Aux \rightarrow does$ [.60] | can [.40]
$VP \rightarrow Verb\ NP$	[.20]	$Preposition \rightarrow from$ [.30] | to [.30]
$VP \rightarrow Verb\ NP\ PP$	[.10]	| on [.20] | $near$ [.15]
$VP \rightarrow Verb\ PP$	[.15]	| $through$ [.05]
$VP \rightarrow Verb\ NP\ NP$	[.05]	
$VP \rightarrow VP\ PP$	[.15]	
$PP \rightarrow Preposition\ NP$	[1.0]	



Probabilistic CYK parsing vs CYK parsing

Book the dinner flight



$P(T_{left}) = .05 * .20 * .20 * .20 * .75 * .30 * .60 * .10 * .40 = \mathbf{2.2 \times 10^{-6}}$

$P(T_{right}) = .05 * .10 * .20 * .15 * .75 * .75 * .30 * .60 * .10 * .40 = \mathbf{6.1 \times 10^{-7}}$

Probabilistic CYK

we	eat	sushi	with	chopsticks
NP 1/4 [0,1]	Ø [0,2]	[0,3]	[0,4]	[0,5]
	V 1 [1,2]	**VP** 1/16 (1/2 * 1 * 1/8) [1,3]	[1,4]	[1,5]
		NP 1/8 [2,3]	[2,4]	[2,5]
			IN 1 [3,4]	[3,5]
				NP 1/8 [4,5]

S	→	NP VP	1
NP	→	NP PP	½
	→	we	¼
	→	sushi	⅛
	→	chopsticks	⅛
PP	→	IN NP	1
IN	→	with	1
VP	→	V NP	½
	→	VP PP	¼
	→	MD V	¼
V	→	eat	1

1
1

Probabilistic CYK parsing vs CYK parsing

What differentiates probabilistic CYK parsing from CYK parsing? Why is this important? How does this affect the algorithms used for parsing?

- CYK Parsing : provide a list of all parses
- P-CYK Parsing: assign scores to parses, show the discrimination between likely and unlikely parses

- Algorithm are very closely related

an	*park*	*by*	*Bob*
[0,1] Det	[0,2] NP	[0,3] -	[0,4] NP, X
	[1,2] N	[1,3] -	[1,4] Y
		[2,3] P	[2,4] PP
			[3,4] NP

[4,5] V	[4,6] -	[4,7] VP	[4,8] -	[4,9] VP, VP
	[5,6] Det	[5,7] NP	[5,8] -	[5,9] NP, X
		[6,7] N	[6,8] -	[6,9] Y
			[7,8] P	[7,9] PP
				[8,9] NP

HMM to CFG

A hidden Markov model assigns each word in a sentence with a tag, e.g.,

Donald/NNP has/VBZ small/JJ hands/NNS

The probability of the sequence is based on the tag-word pairs, and the pairs of adjacent tags. Show how this process can be framed as a CFG, and how various probabilities (e.g., observation, transition, and initial state) can be assigned to productions. What are the similarities and differences between CYK parsing with this grammar, and HMM's Viterbi algorithm for finding the best scoring state sequence?

Hidden Markov Revisiting

What are the parameters do we need to learn when training HMM?

1. Initial state probabilities π
   record the distribution of tags for the first token of each sentence

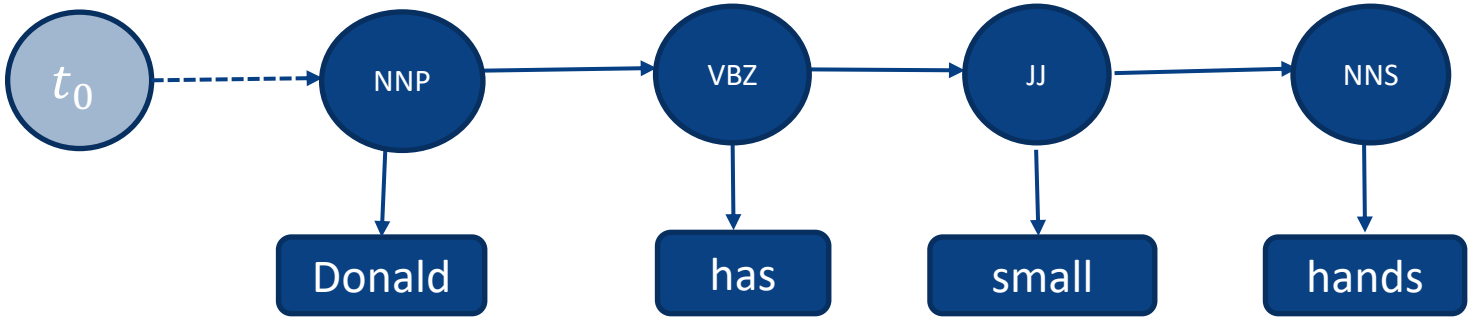
2. Transition probabilities A
   record the distribution of tags of the immediately following token

3. Emission probabilities B
   record the distribution of corresponding tokens

HMM

Donald/NNP has/VBZ small/JJ hands/NNS

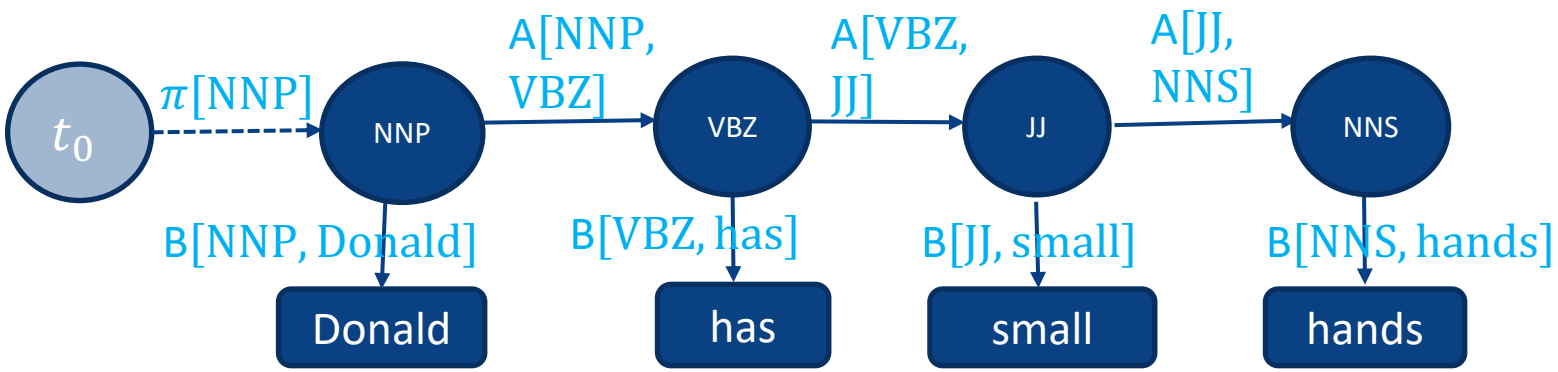
Visualize the HMM



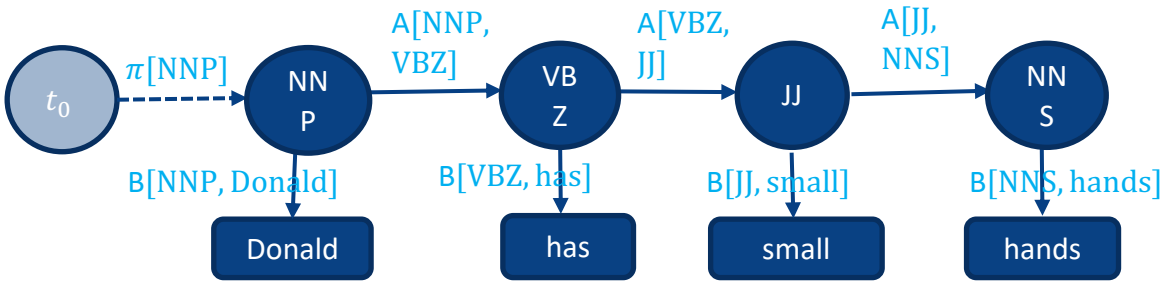
HMM

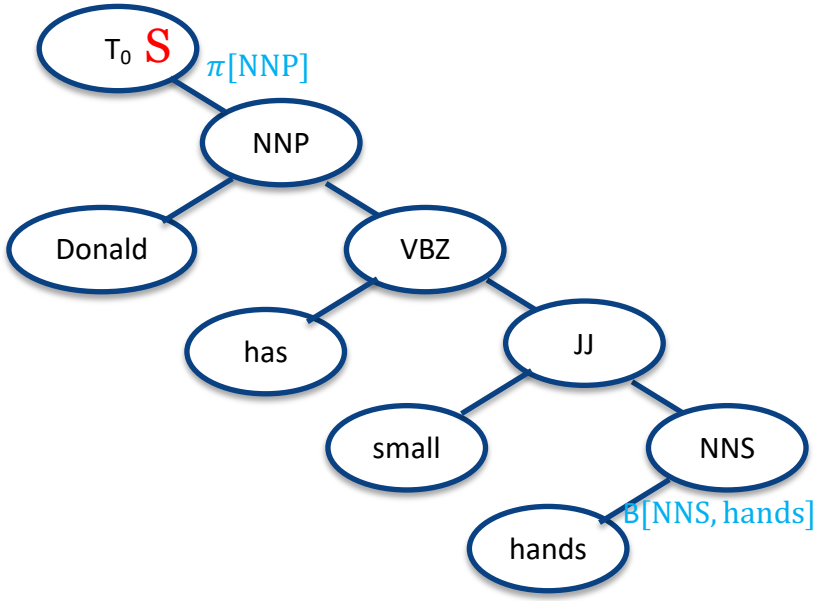
Donald/NNP has/VBZ small/JJ hands/NNS

Visualize the HMM

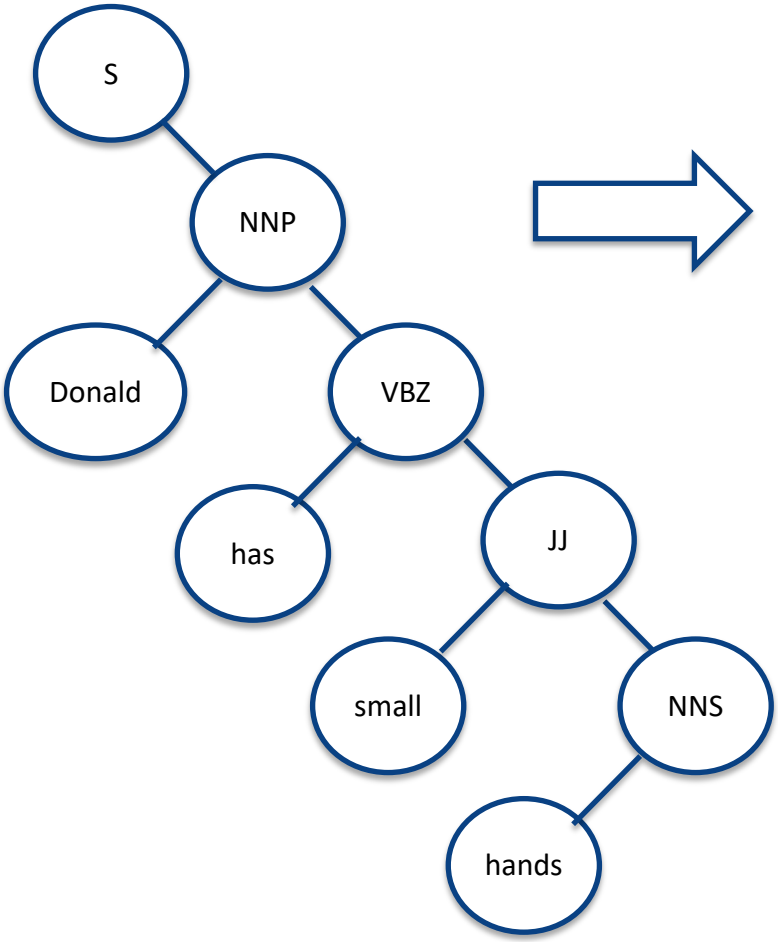


HMM to CFG


t_0
π[NNP]
NNP
A[NNP, VBZ]
VBZ
A[VBZ, JJ]
JJ
A[JJ, NNS]
NNS
B[NNP, Donald]
B[VBZ, has]
B[JJ, small]
B[NNS, hands]
Donald
has
small
hands


T_0 S
π[NNP]
NNP
Donald
VBZ
has
JJ
small
NNS
B[NNS, hands]
hands

HMM



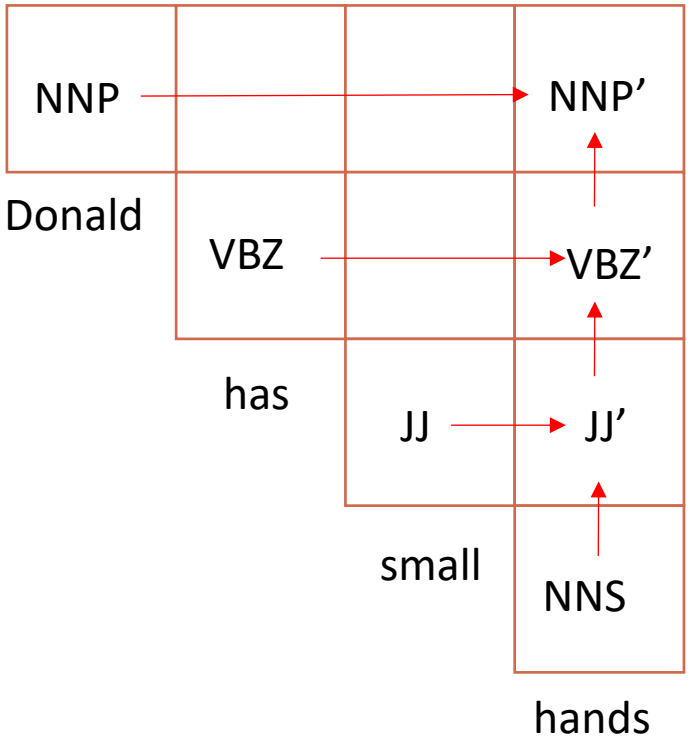
S → NNP
NNP → Donald VBZ.
VBZ → has JJ
JJ → small NNS
NNS → hands

NNP'	→	NNP VBZ'	π[NNP]*A[NNP,VBZ]
VBZ'	→	VBZ JJ'	A[VBZ,JJ]
JJ'	→	JJ NNS	A[JJ,NNS]
NNP	→	Donald	B[NNP,Donald]
VBZ	→	has	B[VBZ,has]
JJ	→	small	B[JJ,small]
NNS	→	hands	B[NNS,hands]

HMM to CFG

Chomsky Normal Form

1. NNP' → NNP VBZ'
2. VBZ' → VBZ JJ'
3. JJ' → JJ NNS
4. NNP → Donald
5. VBZ → has
6. JJ → small
7. NNS → hands



Viterbi V.S. CYK parsing

- CYK parsing under this grammar will return same result as Viterbi in the HMM
- Time complexity?
  - CYK :
  - Viterbi :

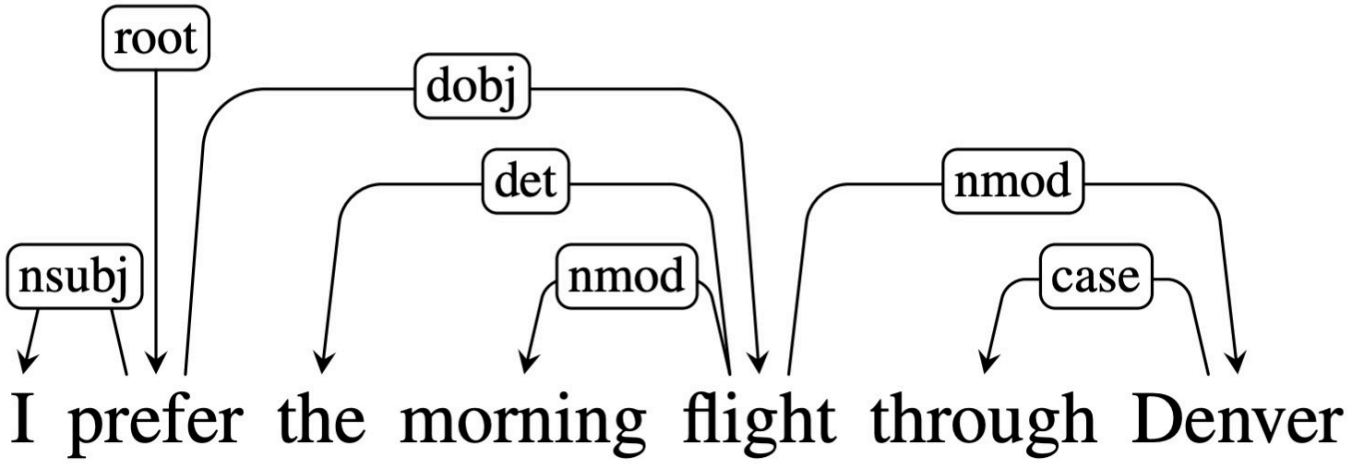
Viterbi V.S. CYK parsing

- CYK parsing under this grammar will return same result as Viterbi in the HMM

- CYK will waste effort assigning analysis to all word spans in the sentence
- Viterbi effectively only considers spans that start at the first word of the sentence.

Dependency Grammars

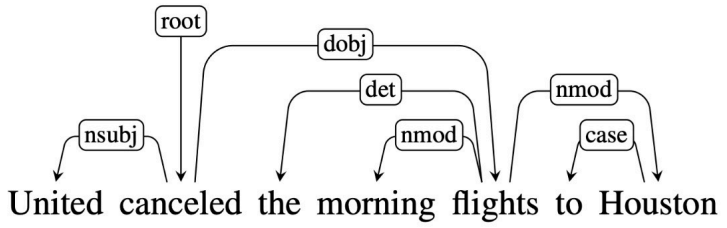
- Describe relations between pairs of words



Dependency Grammars

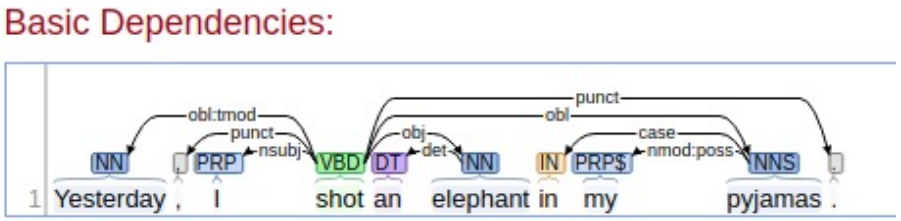
Universal Dependency

Clausal Argument Relations	Description
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction



Yesterday, I shot an elephant in my pyjamas.

Dependency Parsing

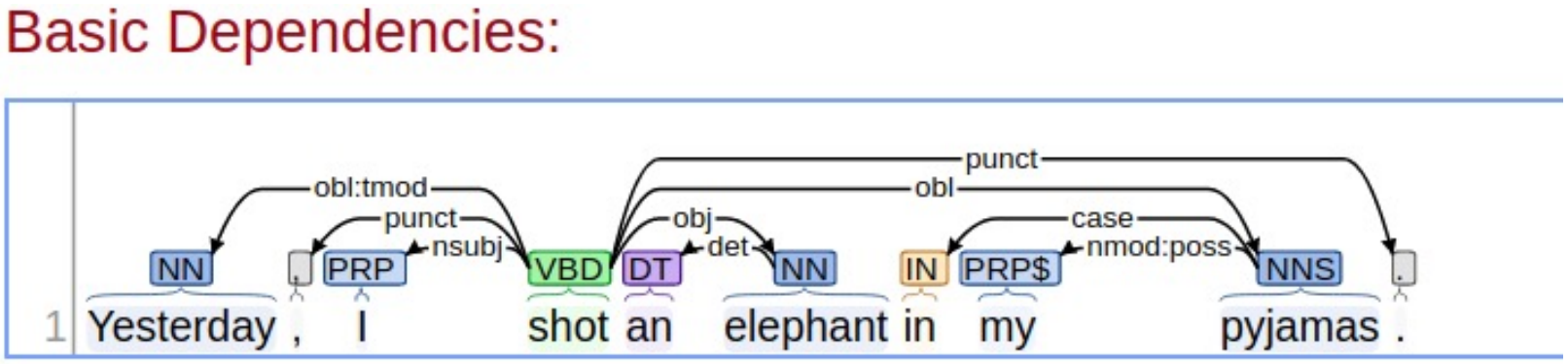


ID	Token	Head	Relation
1	Yesterday	4	OBL:TMOD
2	,	4	PUNCT
3	I	4	NSUBJ
4	shot	0	ROOT
5	an	6	DET
6	elephant	4	DOBJ
7	in	9	CASE
8	my	9	POSS
9	pyjamas	4	OBL
10	.	4	PUNCT

for description of relations: https://universaldependencies.org/u/dep/

Dependency Parsing

Using typical dependency types, construct (by hand) a dependency parse for the following sentence: Yesterday, I shot an elephant in my pyjamas. Check your work against the output of the online GUI for the Stanford Parser: https://corenlp.run/



Transition-based parsing

- Main two data structures :

- Two(Three) types of transitions :

(probabilistic) dependency parsing vs. (probabilistic) CYK parsing

In what ways is (**transition-based, probabilistic**) **dependency parsing** similar to **(probabilistic) CYK parsing**? In what ways is it different?

	Dependency	CYK parsing
determine the structure of a sentence	✓	✓
disambiguate amongst the (perhaps many) possible structures licensed by the grammar using a probabilistic grammar	✓	✓
explicitly tag the sentence		✓
take into account non–local relations in the sentence	✓	
only take into account local relations in the sentence		✓
process the tokens in the sentence one–by–one, left–to– right.	✓	✓
only adds edges that will be in the final structure	✓	
adds numerous fragments which don't end up getting used in the final parse structure		✓

(probabilistic) dependency parsing vs. (probabilistic) CYK parsing

In what ways is (**transition-based, probabilistic**) **dependency parsing** similar to **(probabilistic) CYK parsing**? In what ways is it different?

Similarity:
(1) determine the structure of a sentence;
(2) attempt to disambiguate among the (perhaps many) possible structures licensed by the grammar using a probabilistic grammar;
(3) process the tokens in the sentence one-by-one, left-to-right

(probabilistic) dependency parsing vs. (probabilistic) CYK parsing

In what ways is (**transition-based, probabilistic**) **dependency parsing** similar to **(probabilistic) CYK parsing**? In what ways is it different?

Differences:

(1) Although POS tags are implicitly used in constructing the "oracle" (training), the dependency parser does not explicitly tag the sentence;

(2) the transition-based dependency parser can potentially take into account other (non-local) relations in the sentence, whereas CYK's probabilities only depend on the (local) sub-tree;

(3) CYK add numerous fragments to the chart, not used in the final parse, whereas the transition-based dependency parser only adds edges used in the final structure.