CECZ4045 Natural Language Processing

Parsing – Formal Grammar (Chapter 12)

Syntax

- **The way words are arranged together**
  - Implicit knowledge of your native language that you had mastered by the time you were young without explicit instruction
  - Not the kind of stuff you were later taught in “grammar” school

- Why should we care?
  - Syntax is the key concept for many NLP tasks
    - E.g. information extraction, grammar checking
  - An approximation would be still useful

Key notions

- Constituency
  - Groups of words that behave as a single unit or phrase
  - E.g., a few words form a noun phrase

- Grammatical relations
  - Syntactic relations between constituents
  - E.g., Subject – verb, verb – object

- Subcategorization
  - Certain kinds of relation between words and phrases
  - E.g., *transitive words*, which can take a direct object, and *intransitive words* which cannot take a direct object

Constituent

- **Groups of words that behave as a single unit or phrase**
  - Can all appear in similar syntactic environments

- Example: Noun phrases (NPs) can occur before verbs
  - Three example sentences starting with NPs
    - <u>three parties from Brooklyn </u>*arrive* …
    - <u>a high-class spot such as Mindy's </u>*attracts* …
    - <u>the Broadway coppers </u>*love* …
  - While the whole noun phrase can occur before a verb, it is not true of each of the individual words that make up a noun phrase

Constituent

- Evidence for constituency
  - Stick together through pre-/post-posed constructions (example using a prepositional phrase)
    - On September seventeenth, I'd like to fly from Atlanta to Denver.
    - I'd like to fly on September seventeenth from Atlanta to Denver.
    - I'd like to fly from Atlanta to Denver on September seventeenth.
  - Individual words making of the phrase cannot be placed differently.
    - *On, I'd like to fly from Atlanta to Denver September seventeenth.

- Non-standard constructions (or constituents)
  - I gave John a watch and Mary a doll.
  - *I gave John a watch Mary and a doll.

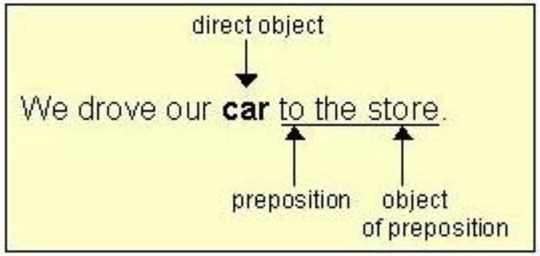
Read Sections 12.2-3 for the following terms

- Noun phrase (NP), verb phrase (VP), prepositional phrase (PP), adjective phrase

- Declarative, imperative, clause
  - Declarative sentences have a subject noun phrase followed by a verb phrase. “I prefer a morning flight”
  - Imperative sentences begin with a verb and have no subject. “Show me the code”.
- Determiner, nominal, quantifier
  - Noun phrases can begin with determiners or quantifier. “a bus stop” “a non-stop flight”
- Auxiliaries, passive, perfect, progressive
  - Auxiliaries include modal verb *can could shall*, perfect auxiliary *have*, progressive auxiliary *be* and passive auxiliary *be*
- Coordination, conjunction
  - A coordinate noun phrase can consist of two noun phrases by a conjunction: “course code and course title”

Grammatical relations

- Syntactic relations **between constituents**
  - Subject-verb
    - E.g. I love him.
  - Verb-object
    - E.g. I love him.
  - Verb-PrepositionalPhrase
    - E.g. I study at the library.
  - Noun-PrepositionalPhrase
    - E.g. The history of Singapore
  - Preposition-object



https://webapps.towson.edu/ows/prepositions.htm

Verb phrases (VPs): Sample patterns

- Verb
  - disappear
  - *prefer
- Verb Object
  - prefer a morning flight
  - *disappear a morning flight
- Verb IObject DObject (indirect object, direct object)
  - give John a watch
  - *prefer John a watch
- Verb Object PrepositionalPhrase(PP)
  - leave Boston in the morning
- Verb PP
  - leave on Thursday

Subcategorization

- ‘Subcategories’ of different types of verbs (by the constituents complement them)
  - Verb "disappear"
  - Verb NP "prefer a morning flight"
  - Verb NP NP "give John a watch"
  - Verb NP PP "leave Boston in the morning"
  - Verb PP "leave on Thursday"

- Subcategorization frame
  - A possible set of complements for a verb
  - E.g. ‘give’: {}, {NP}, {NP, NP}, {NP, PPto}, {PP}
  - Verb: function, complements: arguments
    - e.g. def buy(Person: buyer, Thing: product)

Subcategorization (examples)

- Luke *worked*
  - subcategorization frame contains just a subject argument: **work [NP __]**

- Indiana Jones *ate* chilled monkey brain.
  - The verb eat is transitive, so it subcategorizes for two arguments:
    eat [NP __ (NP)]

- Tom *waited* for us.
  - the verb wait subcategorizes for two arguments as well (although the second of these is an optional prepositional argument):
    wait [NP __ (forNP)]

Exercise: Subcategorization

- Find all subcategorization frames for the verb ‘care’ based on the dictionary definitions

verb [no obj.]

1 [often with negative] feel concern or interest; attach importance to something: *they don't* ***care about*** *human life* | [with clause] *: I don't care what she says.*

• feel affection or liking: *you* ***care*** *very deeply* ***for*** *him.*

• (**care for something/care to do something**) like or be willing to do or have something: *would you care for some tea?* | *I don't care to listen to him.*

2 (**care for**) look after and provide for the needs of: *he has numerous animals to care for.*

Syntactic structure

- Why do we learn about syntax in NLP course?
  - To identify and understand the syntactic structures of sentences

- **Parse tree**
  - Structural representation of grammatical relations expressed in a string

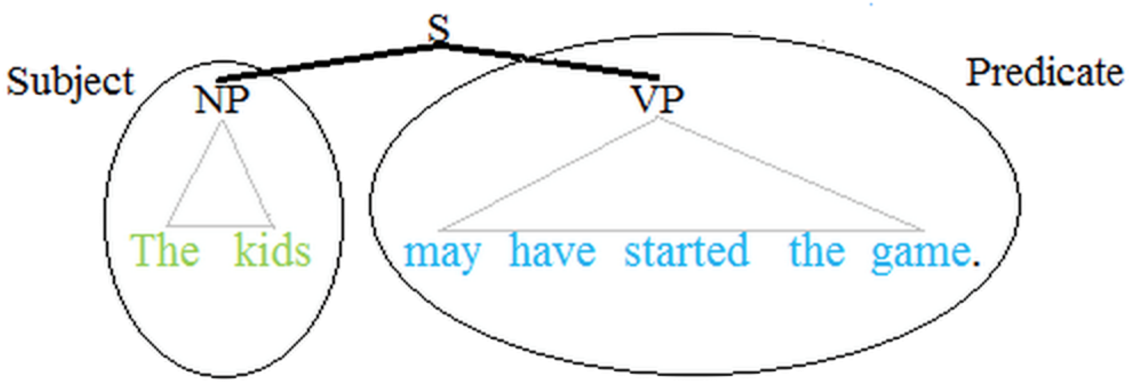
- Representations
  - Phrase structure
  - Dependency structure
  - Predicate-argument structure

Syntactic ≠ Semantic

- iPhone7 is nice but expensive
- iPhone7 is expensive but nice

Phrase Structure

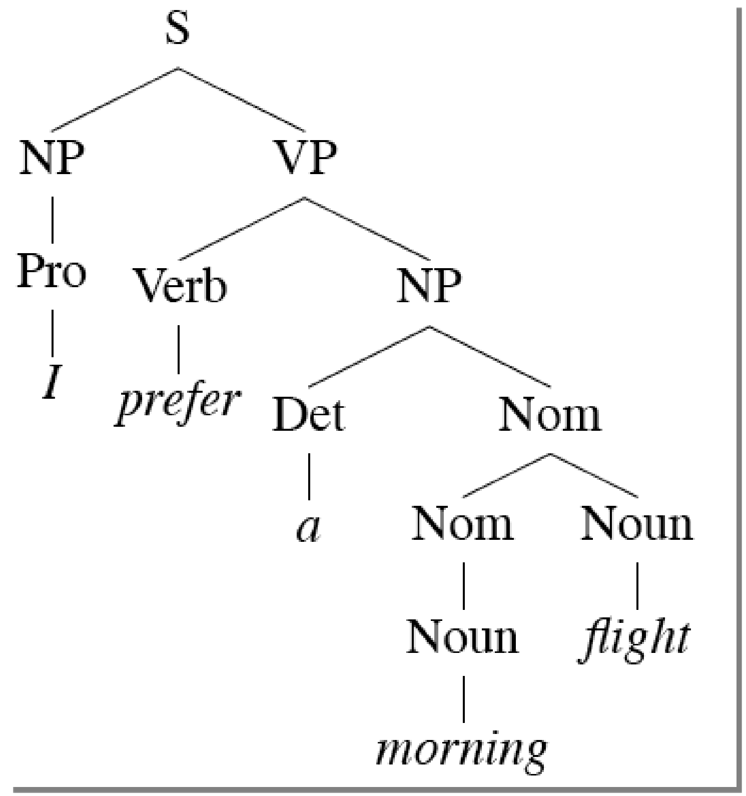
- Phrase structure rules are used to break down a natural language sentence into its constituent parts (also known as syntactic categories), including both lexical categories (parts of speech) and phrasal categories
  - Split a sentence (S) into subject (NP) and predicate (VP)
  - Split subject and predicate into smaller parts
  - Continue until terminal nodes are reached



Phrase Structure (read Section 12.3 for more syntactic tags)

- Organizes words into nested constituents
  - E.g. “I prefer a morning flight”

```
(S
 (NP (Pro I))
 (VP (Verb prefer)
  (NP (Det a)
   (Nom
    (Nom (Noun morning))
    (Noun flight)))))
```



Nominal: (i) noun phrase without determiner; and
(ii) the noun that modifies the head noun in a noun phrase.

Exercise: Phrase structure

- Give the phrase structures of the two sentences:
  - (S1) John gives a present to Mary.
  - (S2) John gives Mary a present.

- POS tags:
  - E.g. ProperNoun: John, Mary
  - E.g. Verb, Det, Noun, Prep

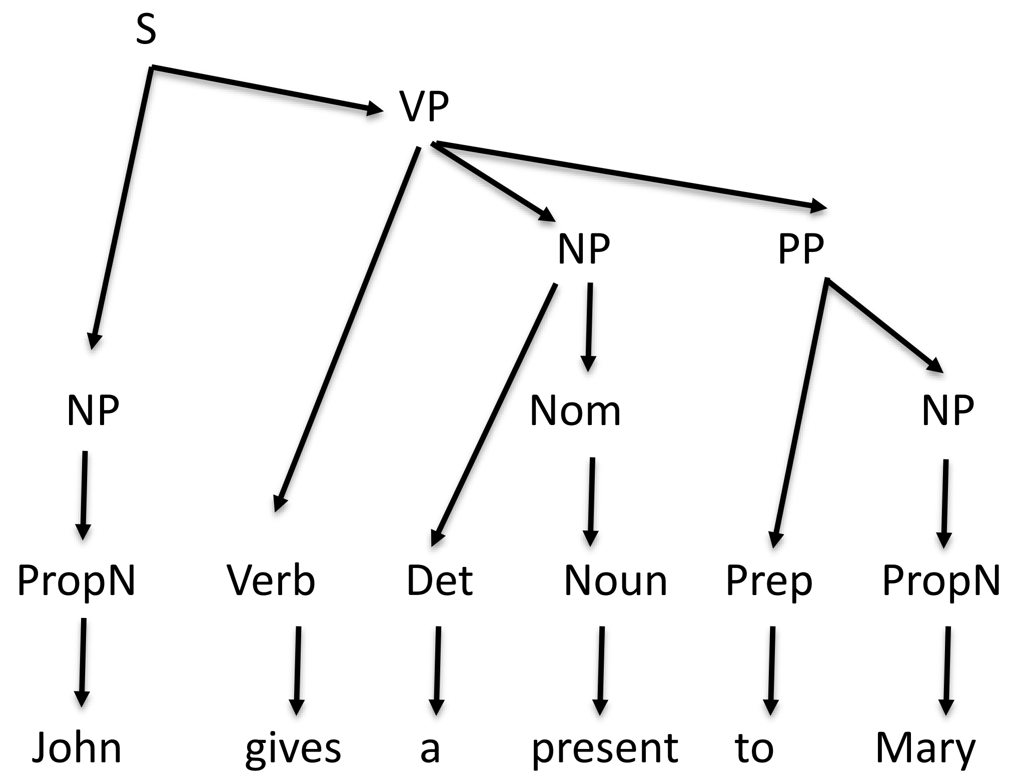
- Example syntactic tags
  - E.g. S, NP, VP, PP_{XX}

Exercise: Phrase structure

- Give the phrase structures of the two sentences:
  - (S1) John gives a present to Mary.
  - (S2) John gives Mary a present.

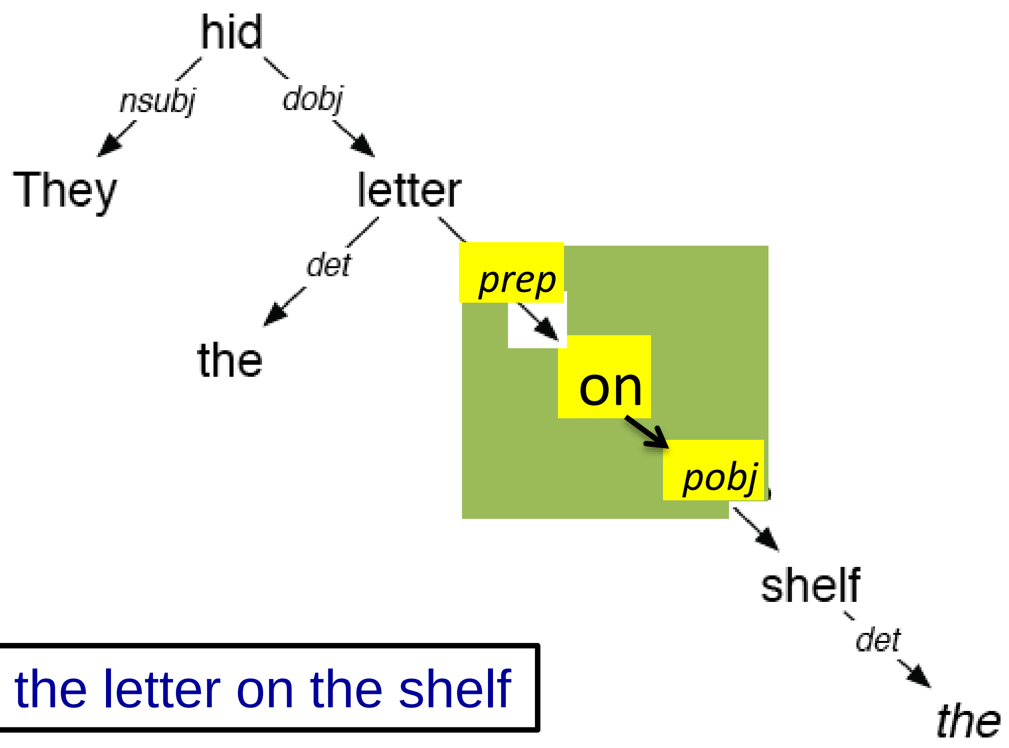
- POS tags:
  - E.g. ProperNoun: John, Mary
  - E.g. Verb, Det, Noun, Prep

- Example syntactic tags
  - E.g. S, NP, VP, PP$_{XX}$



Dependency structure

- Represents grammatical (dependency) relations between pairs of words (i.e. head, dependent)
- Head: Grammatically most important word in a phrase
  - Verb of VP
  - Noun of NP
  - Prep of PP
  - Adj of AdjP
  - …



They hid the letter on the shelf

Dependency Relations (read section 12.7 for more details)

Argument Dependencies	**Description**
nsubj	nominal subject
csubj	clausal subject
dobj	direct object
iobj	indirect object
pobj	object of preposition
Modifier Dependencies	**Description**
tmod	temporal modifier
appos	appositional modifier
det	determiner
prep	prepositional modifier

http://universaldependencies.org/u/dep/index.html

Phrase structure vs. Dependency structure

(S

(NP (Pro **They**))

(VP

(Verb **hid**)

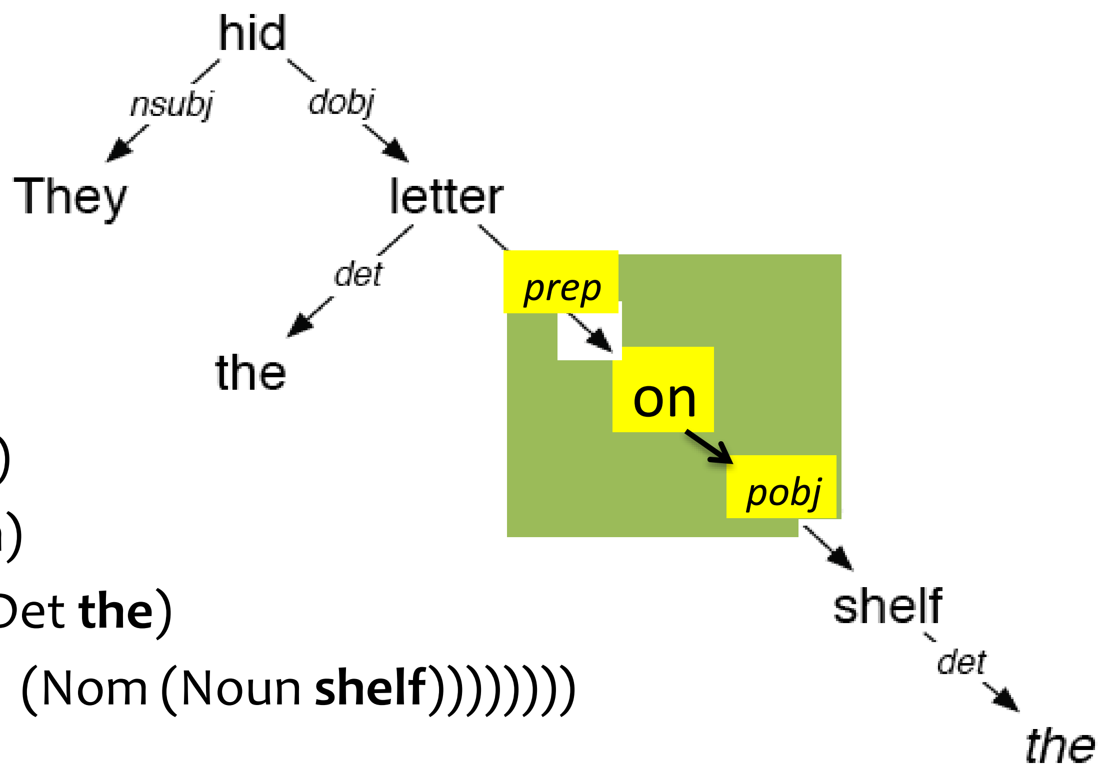
(NP (Det **the**)

(Nom (Noun **letter**)

(PP (Prep **on**)

(NP (Det **the**)

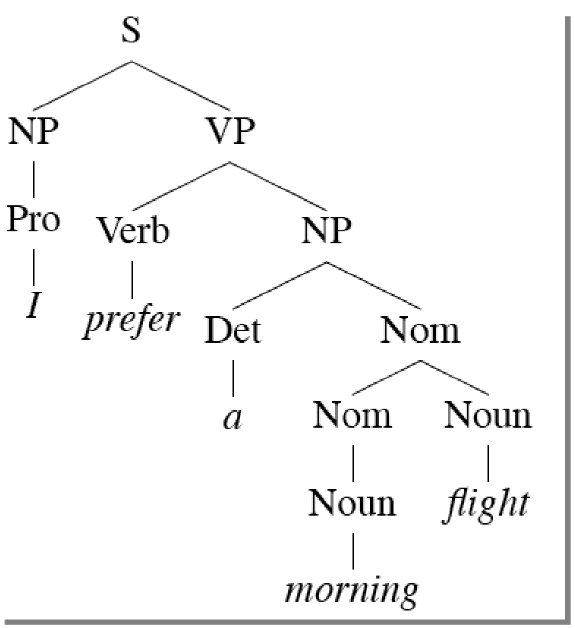
(Nom (Noun **shelf**)))))))



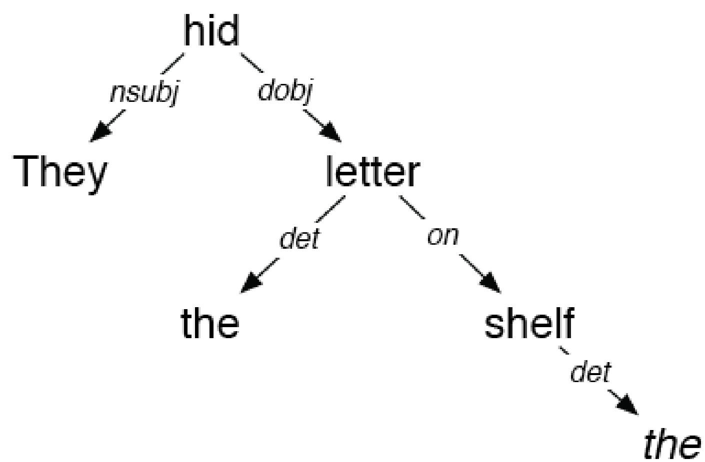
They hid the letter on the shelf

Phrase vs. dependency structure

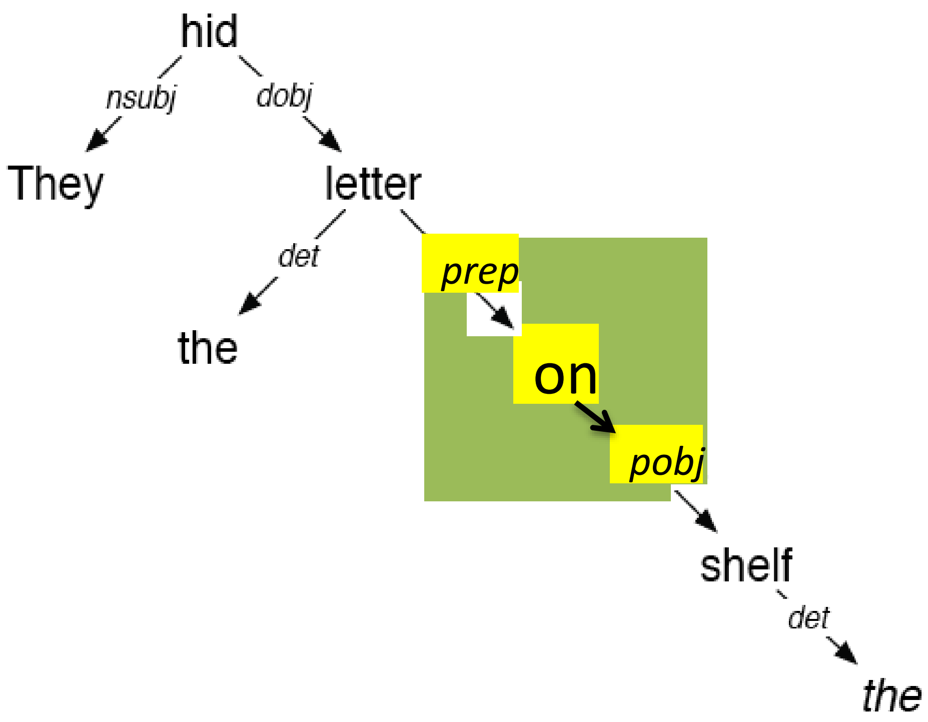
- Constituents are specified
- Grammatical relations are implied
  - Can be deduced from phrase structure

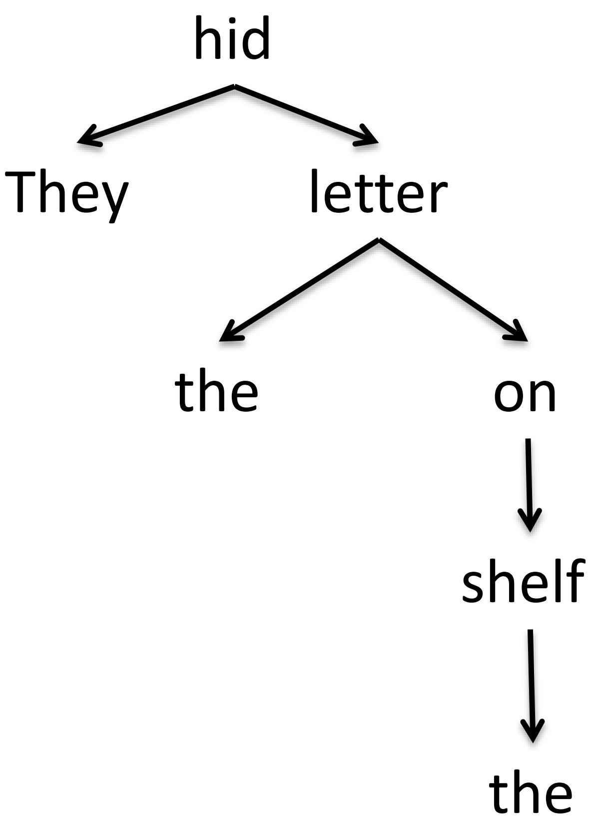


- Grammatical relations are specified
- A subtree corresponds to a constituent
  - Syntactic tag is implied by the root of subtree

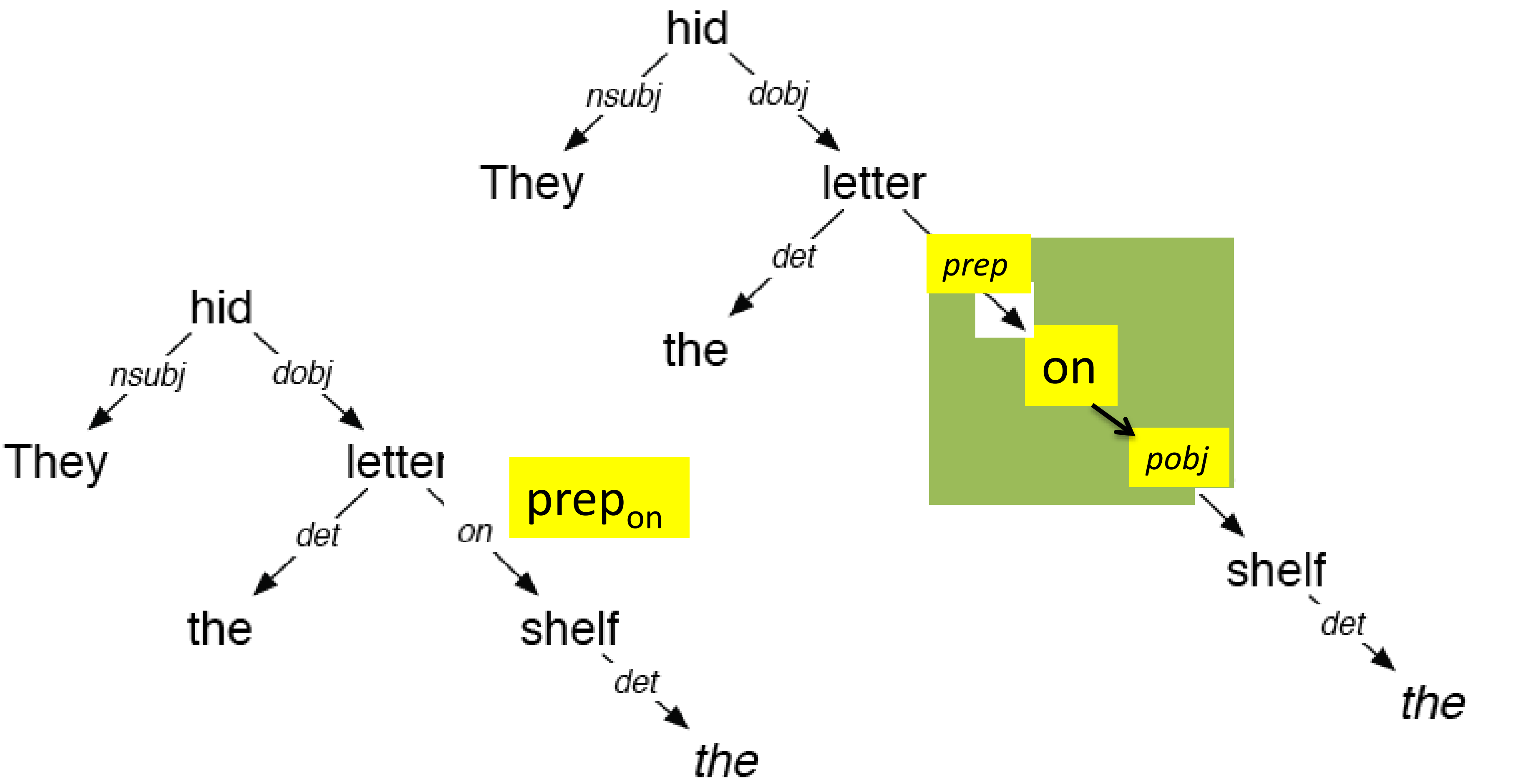


Typed dependency vs. untyped dependency





Typed dependency vs. Collapsed typed dependency



Preposition word followed by its object

Exercise: Dependency structure

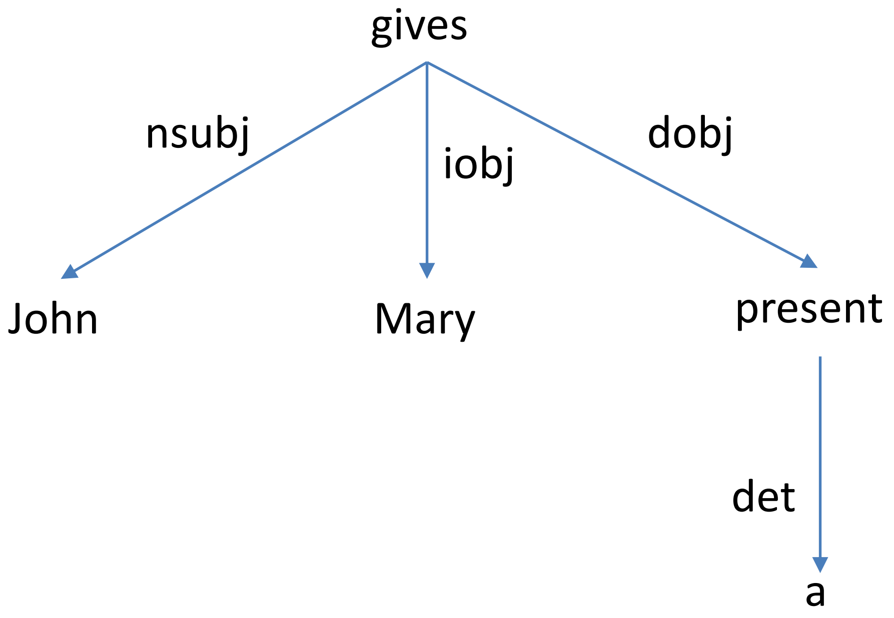
- Give the dependency structures of the two sentences:
  - (S1) John gives a present to Mary.
  - (S2) John gives Mary a present.

- Example dependency relations
  - E.g. nsubj, iobj, dobj, det, prepXX

Exercise: Dependency structure

- Give the dependency structures of the two sentences:
  - (S1) John gives a present to Mary.
  - (S2) John gives Mary a present.

- Example dependency relations
  - E.g. nsubj, iobj, dobj, det, prepXX



Predicate-argument structure

- Review: Subcategorization frame
  - Verb: function/predicate, complements: arguments

- Generalize to all types of words that have complements
  - E.g. Preposition: predicate, object: argument
  - E.g. They hid **the$_1$** letter on **the$_2$** shelf.

Relation type	Predicate	Argument 1	Argument 2
verb_arg12	hid	They	letter
det_arg1	the$_1$	letter	
prep_arg12	on	letter	shelf
det_arg1	the$_2$	shelf	

Dependency structure vs. Predicate-argument structure

- Sentence: They hid the letter on the shelf

Dependency structure	Predicate-argument structure
root(ROOT-0, hid-2) nsubj(hid-2, They-1) dobj(hid-2, letter-4) det(letter-4, the-3) prep_on(letter-4, shelf-7) det(shelf-7, the-6)	verb_arg12(hid-2, They-1, letter-4) det_arg1(the-3, letter-4) prep_arg12(on-6, letter-4, shelf-7) det_arg1(the-6, shelf-7)

Exercise: Predicate-argument structure

- Give the predicate-argument structure of the following sentences
  - John gives Mary a present
  - Mary is easy to please

Example

- John gives Mary a present

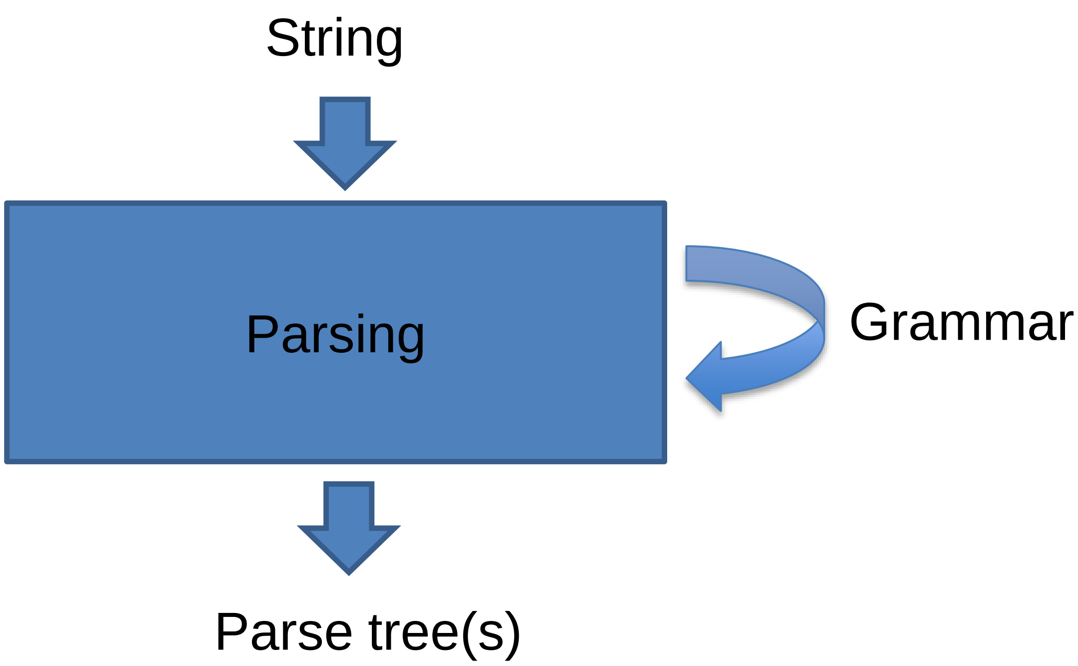
Relation type	Predicate	Argument 1	Argument 2	Argument 3
verb_arg123	gives	John	Mary	present
det_arg1	a	present		

- Mary is easy to please

Relation type	Predicate	Argument 1	Argument 2	
verb_arg12	is	Mary	easy	Subject, description of the subject
adj_arg12	easy	Mary	please	What is easy? In which aspect?
verb_arg12	please		Mary	Who please whom?
comp_arg1	to	please		Complements

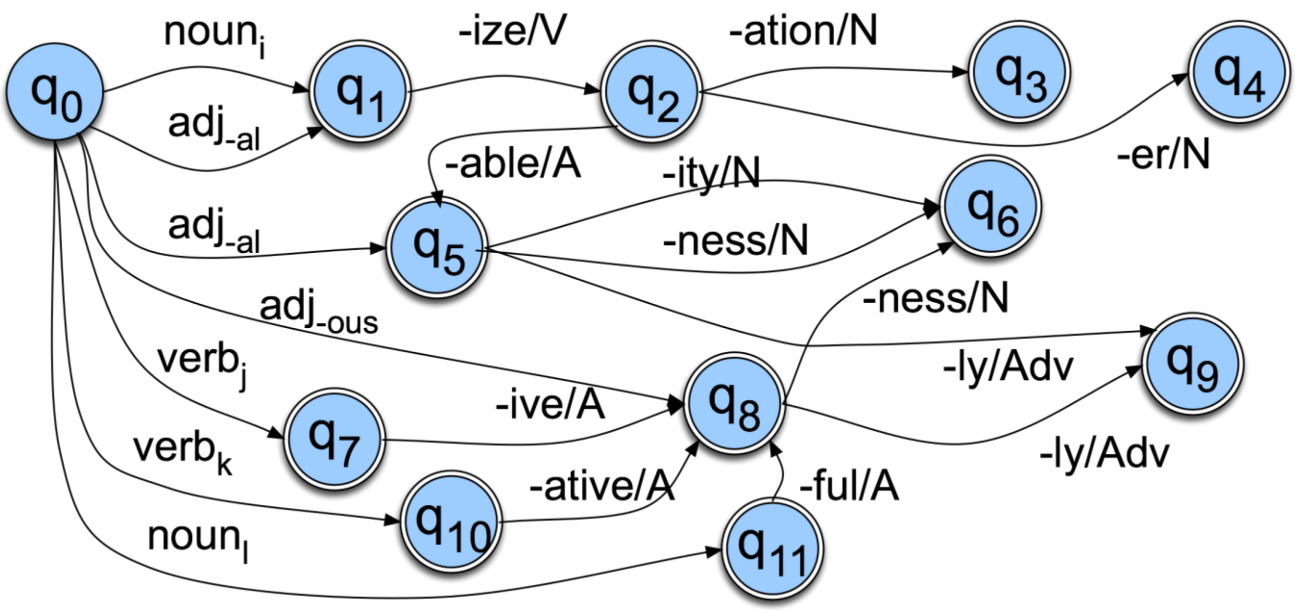
Syntactic parsing

- The process of taking a string and a grammar and returning parse tree(s) for that string



Syntactic parsing

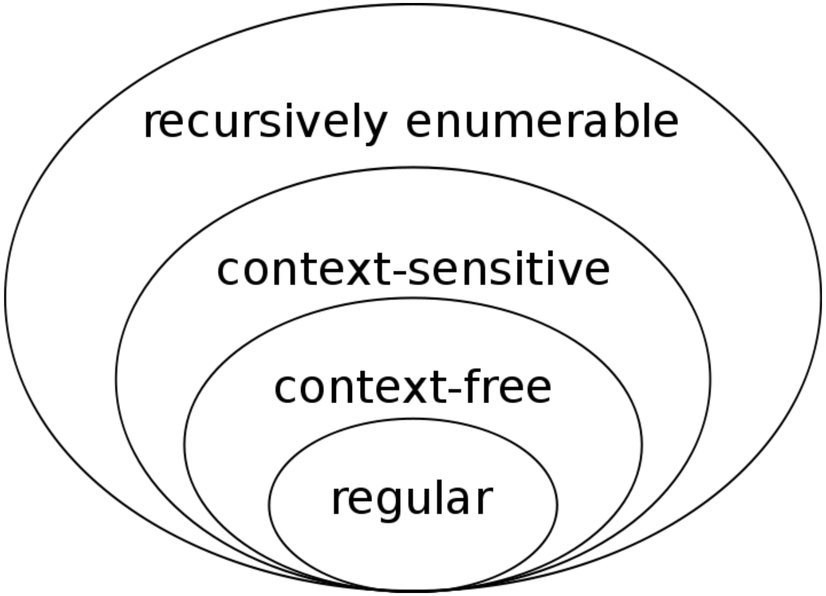
- It is completely analogous to running a finite-state transducer with a string



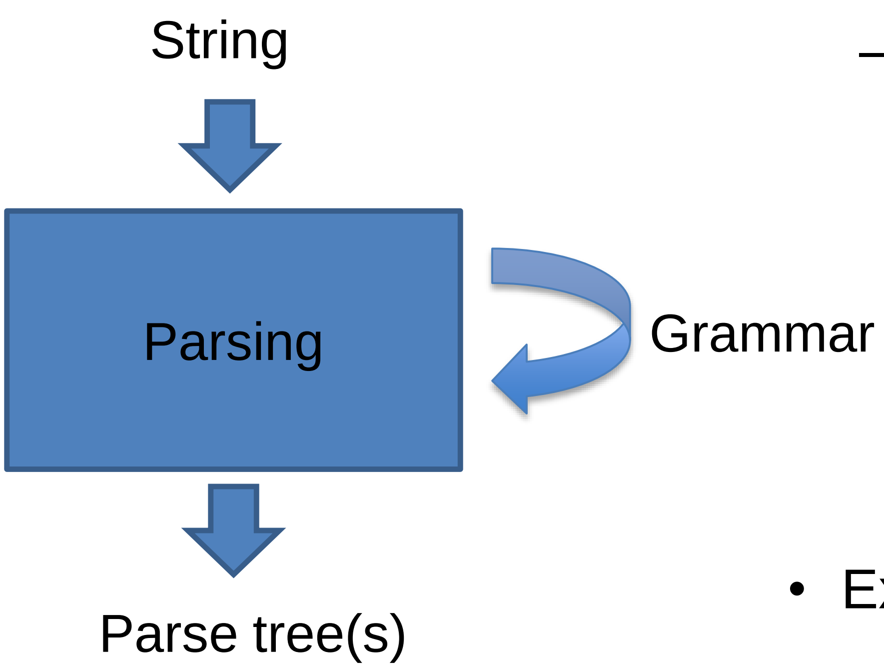
Chomsky hierarchy

- But the grammar for parsing should be more powerful than finite-state transducer

- Chomsky hierarchy (Read Section 16.1 for details)
  - Regular Expressions (RE)
  - Context Free Grammars (CFG)
  - Context Sensitive Grammars (CSG)
  - Unrestricted Grammars (UG)



Syntactic parsing in the course



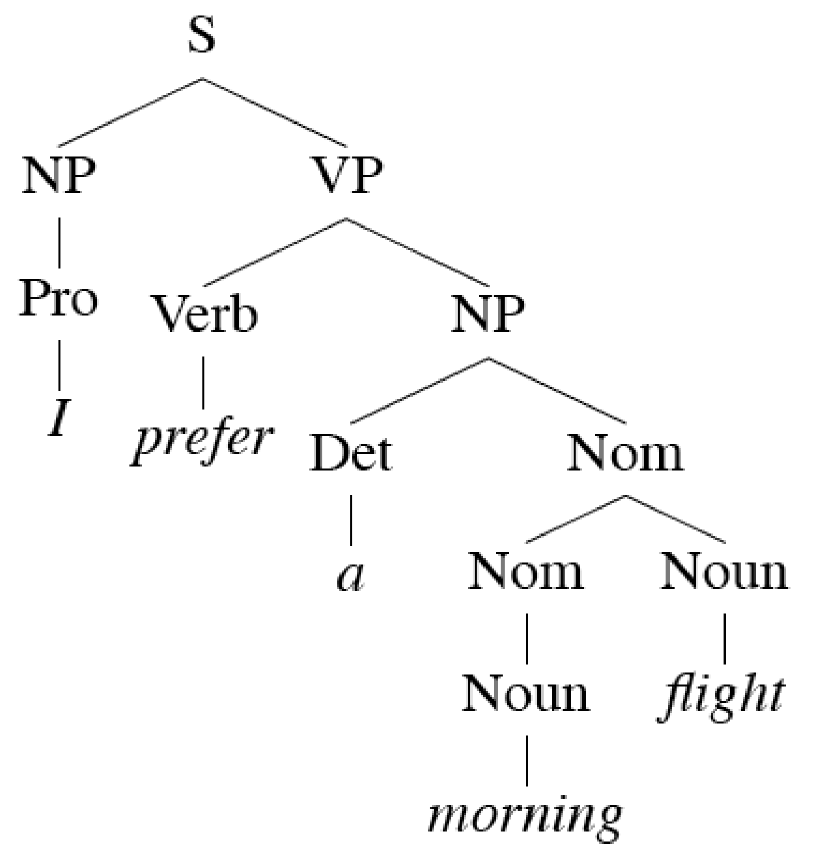
- G = (T, N, S, R)
  - T: a set of terminals (e.g. 'flight')
  - N: a set of non-terminals (e.g. Noun)
  - S: the start symbol, a non-terminal
  - R: rules of the form $X \rightarrow \gamma$
    - X: a non-terminal
    - γ: a sequence of terminals and non-terminals

- Examples on next slide

Context-free grammar (1/2)

- G = (T, N, S, R)
  - T: a set of terminals (e.g. ‘flight’)
  - N: a set of non-terminals (e.g. Noun)
  - S: the start symbol, a non-terminal
  - R: rules of the form X →γ
    - X: a non-terminal
    - γ: a sequence of terminals and non-terminals

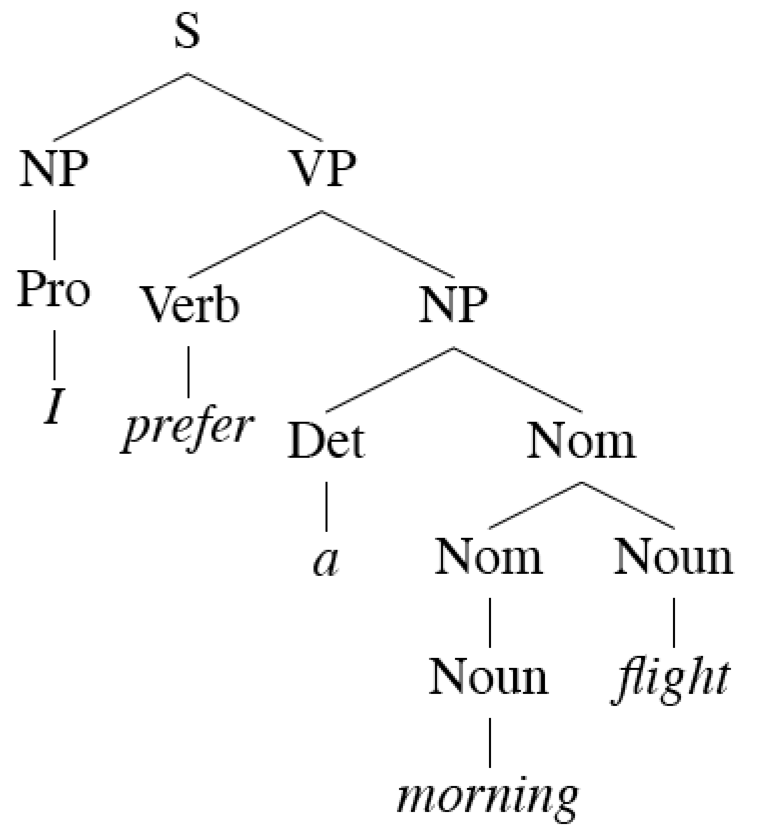
$$
\begin{array}{rcl}
NP & \rightarrow & Det\ Nominal \\
NP & \rightarrow & ProperNoun \\
Nominal & \rightarrow & Noun \mid Nominal\ Noun
\end{array}
$$



Context-free grammar (2/2)

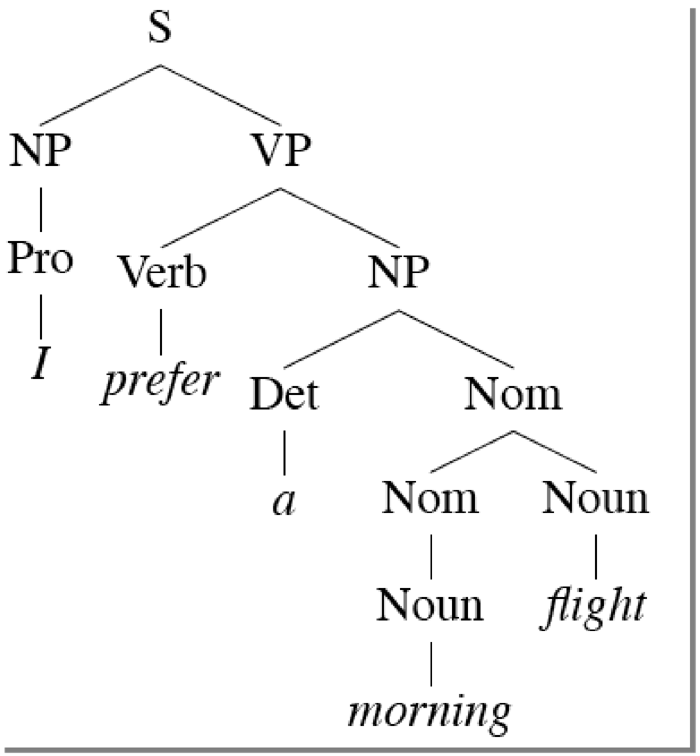
- A grammar G **generates** a language L: **A language is a set of sentences**
- G = (T, N, S, R)
  - T: **words or tokens**
  - N: **POS tags, syntactic tags**
  - S: the start symbol
  - R: rules of the form X →γ
    - X: a non-terminal
    - γ: the sequence of X's children

$$
\begin{array}{rcl}
NP & \rightarrow & Det\ Nominal \\
NP & \rightarrow & ProperNoun \\
Nominal & \rightarrow & Noun \mid Nominal\ Noun
\end{array}
$$



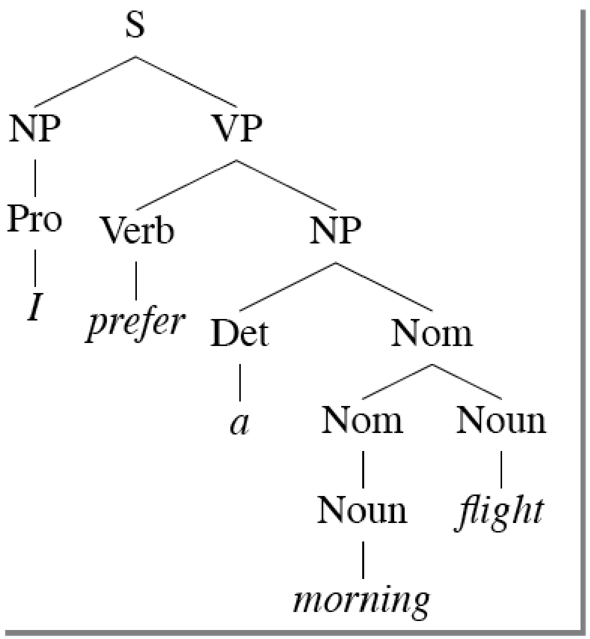
Exercise: Context free grammar

- Give a context free grammar to generate the parse tree



Exercise: Context free grammar

- Give a context free grammar to generate the parse tree
  - T: {I, prefer, a, morning, flight}
  - N: {VP, VP, Pro, Verb, Det, Nom, Noun}
  - S: S
  - R:
    - S -> NP VP
    - NP -> Pro
    - VP -> Verb NP
    - Verb -> prefer
    - NP -> Det Nom
    - Nom -> Nom Noun | Noun
    - Noun -> morning | flight



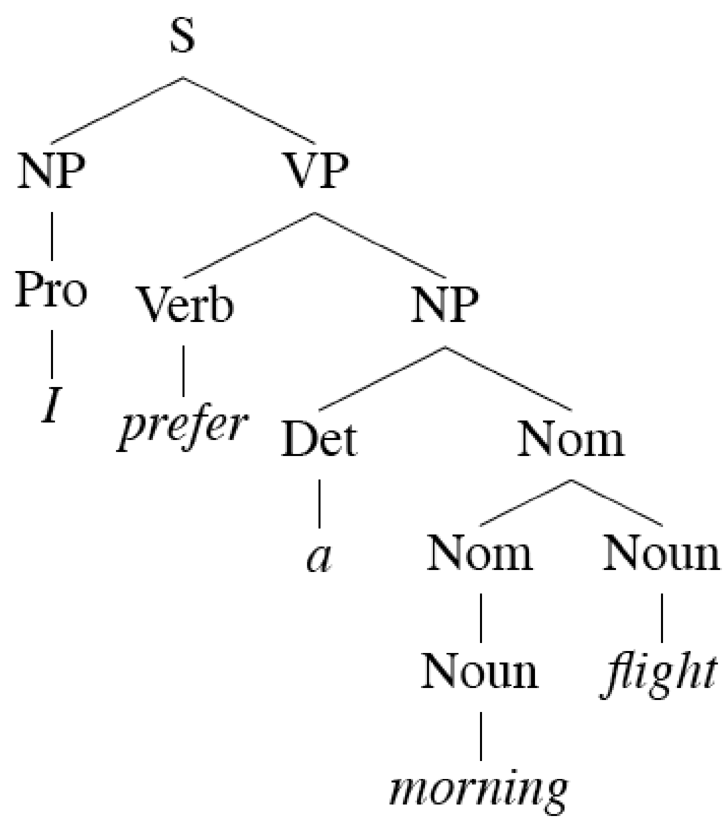
L0 grammar

Grammar Rules			Examples
S	→	*NP VP*	I + want a morning flight
NP	→	*Pronoun*	I
	|	*Proper-Noun*	Los Angeles
	|	*Det Nominal*	a + flight
Nominal	→	*Nominal Noun*	morning + flight
	|	*Noun*	flights
VP	→	*Verb*	do
	|	*Verb NP*	want + a flight
	|	*Verb NP PP*	leave + Boston + in the morning
	|	*Verb PP*	leaving + on Thursday
PP	→	*Preposition NP*	from + Los Angeles

Derivation

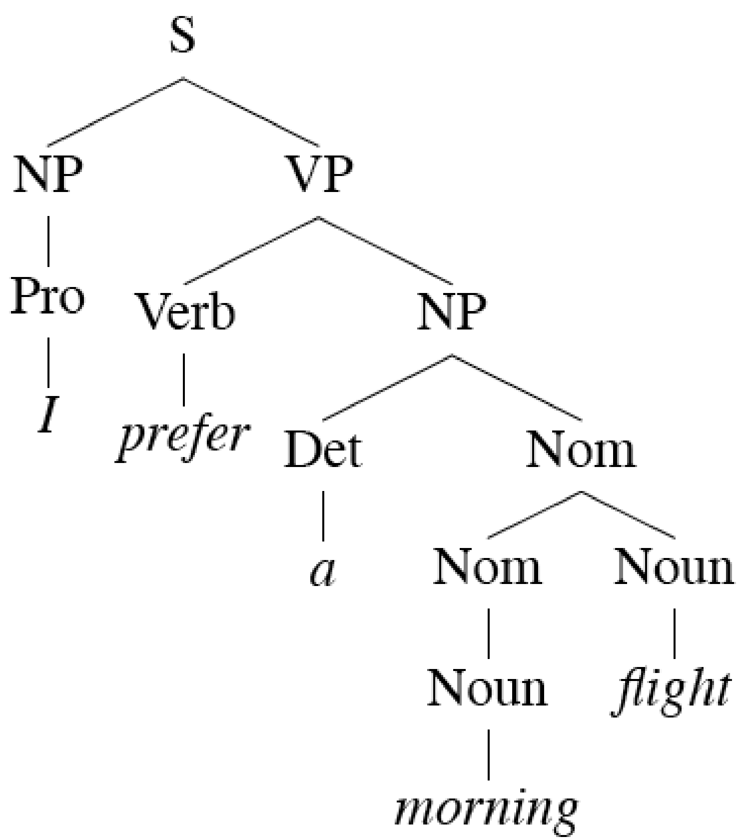
- A sequence of rules applied to a string that accounts for that string

$$
\begin{array}{rcl}
S & \rightarrow & NP\ VP \\
\\
NP & \rightarrow & Pronoun \\
 & | & Proper\text{-}Noun \\
 & | & Det\ Nominal \\
Nominal & \rightarrow & Nominal\ Noun \\
 & | & Noun \\
\\
VP & \rightarrow & Verb \\
 & | & Verb\ NP \\
 & | & Verb\ NP\ PP \\
 & | & Verb\ PP \\
\\
PP & \rightarrow & Preposition\ NP
\end{array}
$$



I prefer a morning flight.

Derivation (2)



S → NP VP
NP → Pro
Pro → ***I***
VP → Verb NP
Verb → ***prefer***
NP → Det Nom
Det → ***a***
Nom → Nom Noun
Nom → Noun
Noun → ***morning***
Noun → ***flight***

Summary: CFG & phrase structure

- G = (T, N, S, R)
  - Terminals: **words**
  - Non-terminals: **constituent names** (e.g. NP, VP)
  - Start symbol: S (sentence) or NP (noun phrase)
  - Rules: e.g. subcategorization frames for verbs

- Derivation will generate the phrase structure of an input

Example grammar formalisms

- Context free grammar (CFG)
- Combinatory categorial grammar (CCG)
- Dependency grammar (DG)

- HPSG: head-driven phrase structure grammar
- LFG: lexical functional grammar
- TAG: tree-adjoining grammar

Combinatory categorial grammar

- An extension of categorial grammar
- A lexicalized grammar model, consisting of
  - A categorial lexicon and
  - (a few) combinatory rules (or operators)

- Cf. CFG
  - POS tags for words (e.g. Noun, Verb)
  - (a lot of) rules (e.g. S $\rightarrow$ NP VP)

Categorial lexicon

- Operators
  - X/Y: a function that combines with a Y on its **right** to produce an X
  - X\Y: a function that combines with a Y on its **left** to produce an X

- Example categories
  - Determiner: NP/N (e.g. <u>the</u> boy)
  - Transitive verb: (S\NP)/NP (e.g. Harry <u>eats</u> apples)

  - The role of determiner is to make a noun on its right into a noun phrase
  - A transitive verb requires a noun phrase to its left and a noun phrase to its right to make a sentence

Derivation of CCG

Harry	**eats**	**apples**	
NP	(S\NP)/NP	NP	>
	S\NP		<
	S		

Application combinators

$$\frac{\alpha : X/Y \qquad \beta : Y}{\alpha\beta : X} >$$

$$\frac{\beta : Y \qquad \alpha : X\backslash Y}{\beta\alpha : X} <$$

We focus on CFG!

- Classic approach
  - Still widely used for practical parsing systems
- Theoretically well-studied
  - Equivalent to Backus-Naur Form (BNF)
  - SQL is formally defined in BNF

Summary

- Syntax
  - Key notions
    - Constituency
    - Grammatical relations
    - Subcategorization
  - Syntactic structure
    - Phrase structure
    - Dependency structure
    - Predicate-argument structure
  - Syntactic grammar
    - Context free grammar
    - Dependency grammar
    - Combinatory categorial grammar