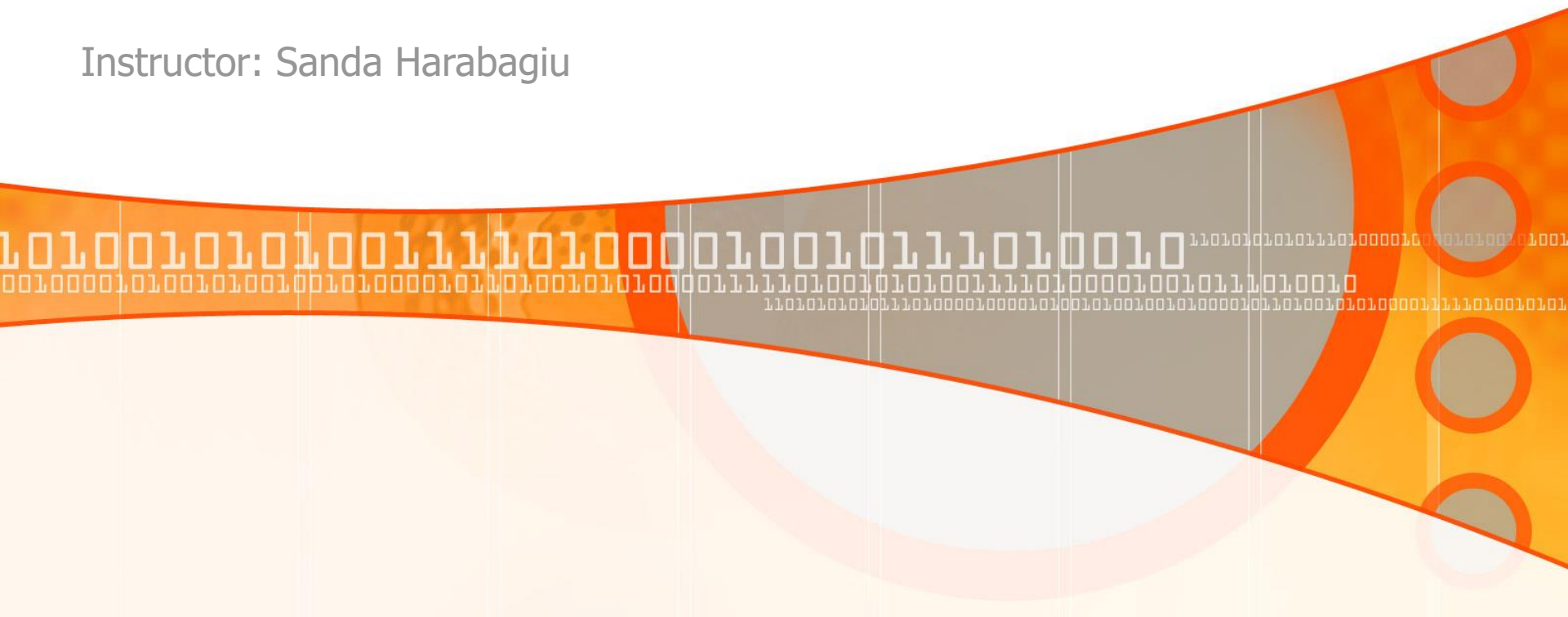
Natural Language Processing
CS 6320

Lecture 9

Formal Grammars of English

Instructor: Sanda Harabagiu



Introduction into Syntax

- *We introduce some sophisticated notions:*
  - **Constituency**
    - Groups of words may behave as a unit – called a *constituent*
  - **Grammatical relations**
    - Subjects and objects
  - **Sub-categorization and dependency**
    - Certain kinds of relations between words and phrases

- **Formal Grammars**:
  - Context-free grammars
  - Dependency Grammars
  - Categorial Grammar

Constituency 1/2

- An example of how words are grouped in English: the **Noun Phrase** (NP)
- NP – a group of words surrounding at least one noun. Examples:

Harry the Horse	A high-class spot such as Mindy's
The Broadway coopers	The reason he comes into the Hot Box
they	Three parties from Brooklyn

- *How do we know that they group together???* They appear in similar syntactic contexts, e.g. before a verb:

Three parties from Brooklyn *arrive*
A high-class spot such as Mindy's *attracts*
The Broadway coopers *love*
they *sit*

- But not all words from an NP can appear before verbs, e.g.:

*from *arrive*	*as *attracts*
*the *is*	*spot *is*

Constituency 2/2

- Other forms of evidence:
  - Preposed and postposed constructions

On September seventeenth, I'd like to fly from Atlanta to Denver.	← Preposed
I'd like to fly from Atlanta to Denver *on September seventeenth.*	← Postposed
I'd like to fly *on September seventeenth* from Atlanta to Denver.	

- *How the entire phrase can be placed differently, but the words cannot:*

**On September* I'd like to fly *seventeenth* from Atlanta to Denver.
**On* I'd like to fly *September seventeenth* from Atlanta to Denver.
I'd like to fly *on September* from Atlanta to Denver *seventeenth*.

Context Free Grammars

- *The most widely used formal system for modeling constituent structure in English and other natural languages is the **Context-Free Grammar**, or CFG. Context-free grammars are also called Phrase-Structure Grammars, and the formalism is equivalent to Backus-Naur Form, or BNF*
- A CFG consists of
  1. *rules* or *productions* – expressing the way in which symbols of the language can be grouped together, and
  2. a *lexicon* of words and symbols.
- *Example:* the NP

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

Productions

$$
\begin{aligned}
Det &\rightarrow a \\
Det &\rightarrow the \\
Noun &\rightarrow flight
\end{aligned}
$$

Lexicon

Symbols in CFGs

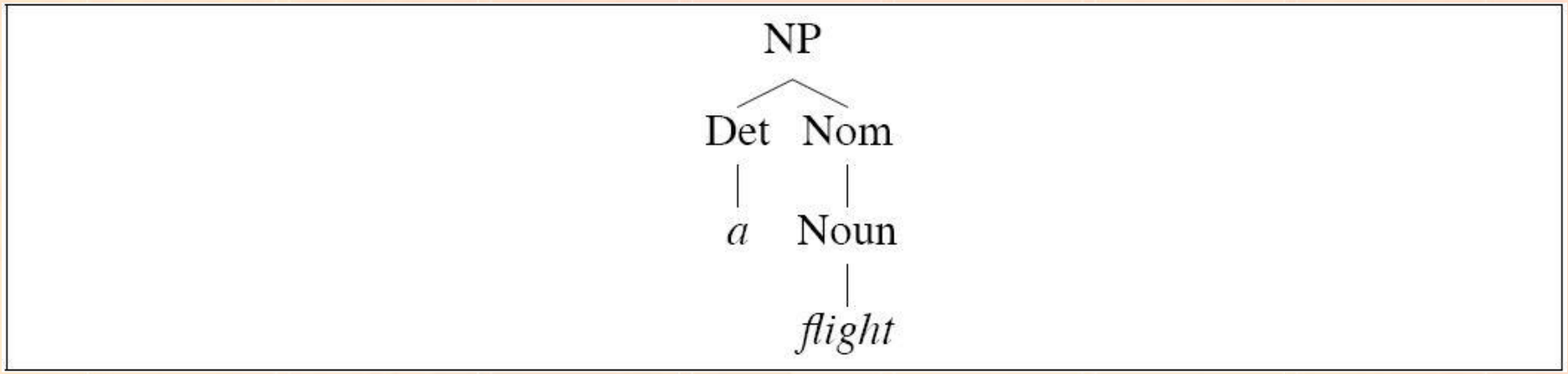
- *The symbols that are used in a CFG are divided into two classes:*

1. *The symbols that correspond to words in the language (“the”, “nightclub”) are called **terminal symbols**; the lexicon is the set of rules that introduce these terminal symbols.*
2. *The **non-terminal symbols** that express abstractions over the terminals. In each context-free rule, the item to the right of the arrow (→) is an ordered list of one or more terminals and non-terminals; to the left of the arrow is a single non-terminal symbol expressing some cluster or generalization.*

$$\begin{aligned} Det &\rightarrow a \\ Det &\rightarrow the \\ Noun &\rightarrow flight \end{aligned}$$

Derivations in Context Free Grammars

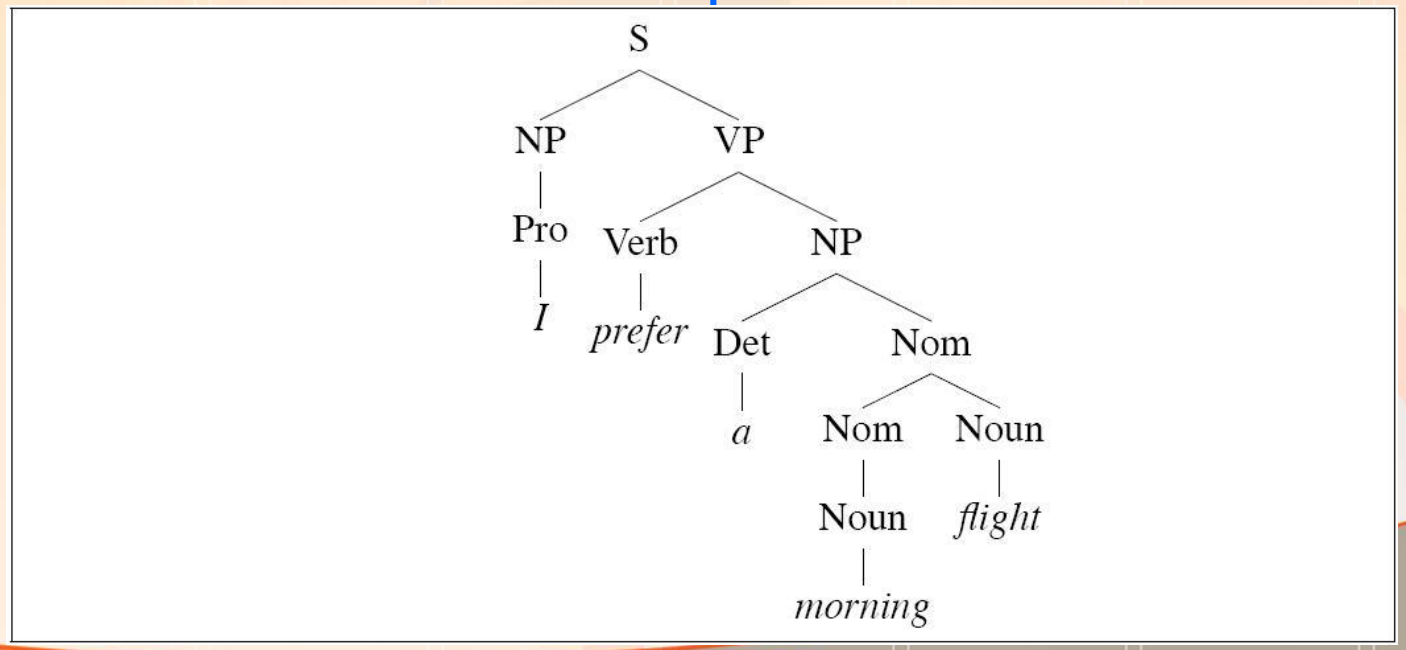
- A CFG can be thought of in two ways
  - A device for *generating sentences*
  - A device for *assigning structure* to a sentence



Parse tree derived for the string "a flight"

Rules in Context Free Grammars

- A CFG has a start symbol S
- Rules that expand S are called **high-level rules**
  - S → NP VP *I prefer a morning flight*
  - VP → Verb NP *prefer a morning flight*
  - VP → Verb NP PP *leave Boston in the morning*
  - VP → Verb PP *leaving on Thursday*
  - PP → Preposition NP *from Los Angeles*



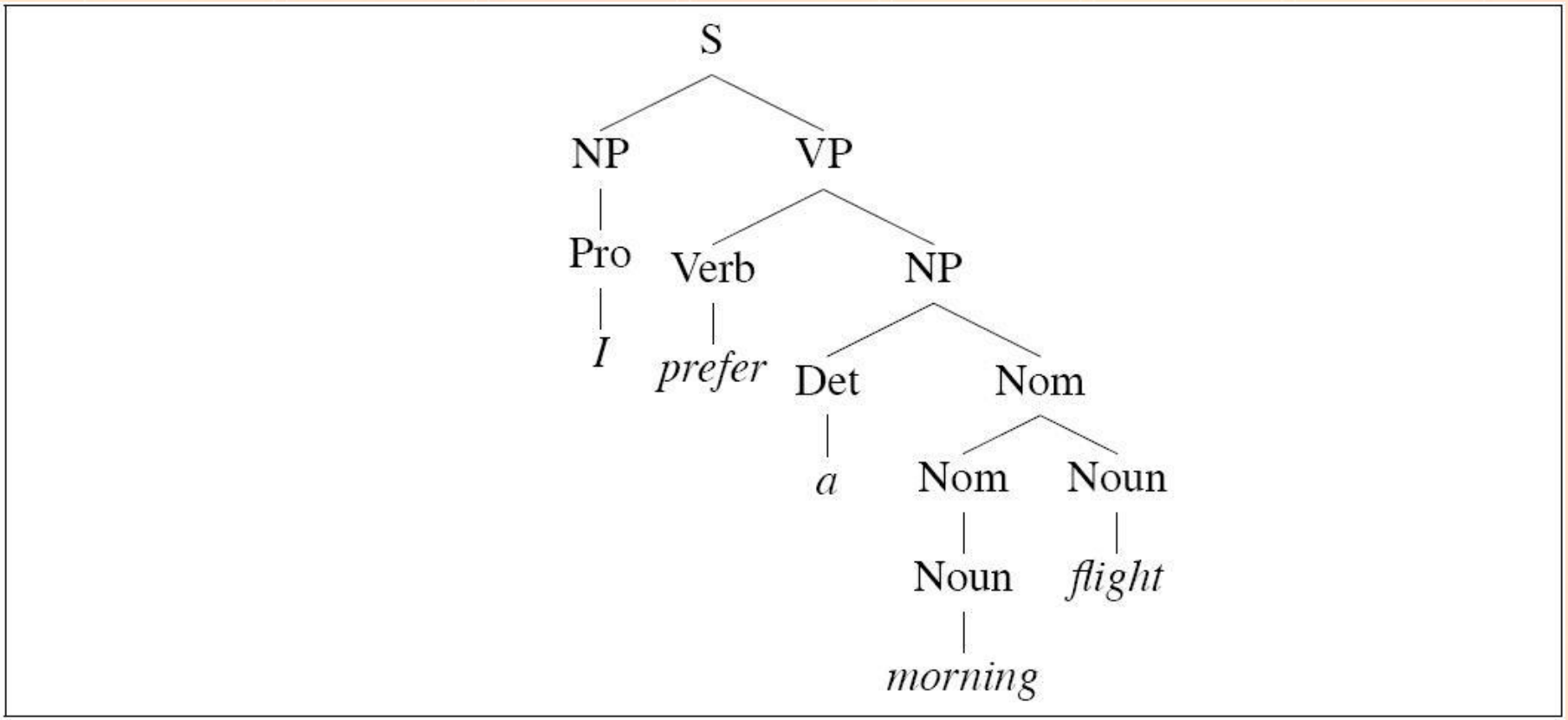
Grammar for $\mathcal{L}_0$

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

Lexicon for $\mathcal{L}_0$

Noun →	*flights* | *breeze* | *trip* | *morning*
Verb →	*is* | *prefer* | *like* | *need* | *want* | *fly*
Adjective →	*cheapest* | *non-stop* | *first* | *latest* | *other* | *direct*
Pronoun →	*me* | *I* | *you* | *it*
Proper-Noun →	*Alaska* | *Baltimore* | *Los Angeles* | *Chicago* | *United* | *American*
Determiner →	*the* | *a* | *an* | *this* | *these* | *that*
Preposition →	*from* | *to* | *on* | *near*
Conjunction →	*and* | *or* | *but*

Parse trees and bracketed notations



- Can be represented in a more compact way – bracketed:

$[_S\ [_{NP}\ [_{Pro}\ I]]\ [_{VP}\ [_V\ prefer][_{NP}\ [_{Det}\ a]\ [_{Nom}\ [_N\ morning]\ [_{Nom}\ [_N\ flight]]]]]]$

Formal definition of CFG

- **A CFG has four parameters** (a 4-tuple):

1. A set of non-terminal symbols N
2. A set of terminal symbols Σ
3. A set of productions P, each of them of the form $A \rightarrow \alpha$ where A is a *non-terminal* and α is a *string of symbols* from the infinite set of strings $(\Sigma \cup N)^*$
4. A designated start symbol S

- *A language is defined via the concept of* **derivation**.

- If $A \rightarrow \beta$ is a production from P, and α and γ are strings from $(\Sigma \cup N)^*$, we say that $\alpha A \gamma$ **directly derives** $\alpha\beta\gamma$ or $\alpha A \gamma \Rightarrow \alpha\beta\gamma$
- Derivation is a generalization of direct derivation:
  - Let $\alpha_1\ \alpha_2\ ,\ldots\ \alpha_m$ be strings in $(\Sigma \cup N)^*$, $m \geq 1$, such that
  - $\alpha_1 \Rightarrow \alpha_2$, $\alpha_2 \Rightarrow \alpha_3$ and $,\ldots,\ \alpha_{m-1} \Rightarrow \alpha_m$
  - We say that α_1 derives α_m or $\alpha_1 \overset{*}{\Rightarrow} \alpha_m$
- $\mathcal{L}_G = \{w \mid w \text{ is in } \Sigma^* \quad S \overset{*}{\Rightarrow} w\}$

Chomsky's Classification

- Chomsky identifies four classes of grammars:
  - **Class 0**: unrestricted phrasestructure grammars. No restriction on type of rules.
  - **Class 1**: context sensitive grammars.

    $$x \rightarrow y$$

    x, y are sequences of terminals and/or nonterminal symbols. The length of y is greater or equal to the length of x.
  - **Class 2**: Context free grammars

    $$A \rightarrow x$$

    A is a nonterminal.

    x is a sequence of terminals and/or nonterminal symbols.
  - **Class 3**: regular grammars

    $A \rightarrow Bt$ or

    $A \rightarrow t$

    A, B are nonterminals

    t is a terminal.

Note: The higher the class the more restrictive it is.

Sentence-Level Constructions 1/3

- Sentences have numerous constructions, but four are particularly common:

1. Declarative structure
2. Imperative structure
3. Yes-no question structure
4. Wh-question structure

- Sentences with *declarative structure* (e.g. "I prefer morning flights") have a subject NP followed by a VP.
- Examples:
  - *The flight should be eleven a.m. tomorrow.*
  - *The return flight should leave around seven p.m.*
  - *I'd like to fly the coach discount class.*
  - *I want a flight from Ontario to Chicago.*
- We can model this structure with a rule: S → NP VP

Sentence-Level Constructions 2/3

- Sentences with *imperative structure* (e.g. "Show the lowest fare.") often begin with a VP, and have no subject.
- Examples:
  - *Show me the cheapest fare that serves lunch.*
  - *List all the flights between five and seven p.m.*
  - *Give me Sunday's flights arriving in Las Vegas from New York City.*
- We can model this structure with a rule: S → VP
- Sentences with *yes-no question structure* (e.g. "Do any of these flights have stops ?") is used to ask questions. They typically begin with an auxiliary verb followed by a subject NP , followed by a VP.
- Examples:
  - *Does American flight eighteen twenty five serve dinner?*
  - *Can you give me the same information on United?*
- We can model this structure with a rule: S → Aux NP VP

Sentence-Level Constructions 3/3

- Sentences with *wh-structure* (e.g. "What airlines fly from Burbank to Denver ?") have one constituent a **wh-phrase**, i.e. a phrase having a wh-word (who, when, whose, where, what, which, how, why).
- We have a a *wh-subject question structure* and a *wh-non-subject question structure*. The wh-subject structure is identical to the declarative structure, except that the first NP contains some wh-word.
  - Examples:
    - *Which flights depart Burbank after noon and arrive in Denver by six p.m.?.*
    - *Whose flights serve breakfast?*
    - *Which of these flights have the longest layover in Nashville?*
  - We can model this structure with a rule: S → Wh-NP VP
- In sentences with *wh-non-subject question structure* the wh-phrase is not the subject of the sentence, thus the sentence includes another subject. In suhc sentences, the auxiliary appears before the subject NP, like in the yes-no-question structures.
  - Examples:
    - *What flights do you have from Burbank to Tacoma Washington?*
  - We can model this structure with a rule: S → Wh-NP Aux NP VP

The Noun Phase

- Three of the most popular NPs:

- NP → Pronoun
- NP → Proper Noun
- NP → Det Nominal | Nominal → Nominal Noun | Nominal → Noun

NP → Modifiers HEAD-Noun Modifiers

- **The determiner**:
- The role can be filled by simple lexical determiners or by more complex expressions(e.g. possessive expressions)

Examples 1: a flight | this flight | any flights | those flights | some flights

Examples 2: United's flight |United's pilot's union | Denver's mayor's mother's canceled flight

Possessive expressions are defined by: Det → NP's

- **The nominal:**
- Can be either a simple noun or a construction in which a noun (Nominal → Noun) is in the center and it also have pre- and post-head modifiers.

Before the Head Noun

- Several word classes may appear before the head: cardinal numbers, ordinal numbers and quantifiers.
- Examples: cardinal numbers – two friends | one stop
- Examples: ordinal numbers – the first friend | the next stop | the other flight
- Examples: quantifiers – many friends | several stops | few flights | much noise
- Adjectives occur after quantifiers but before nouns
- Adjectives can also be grouped into an adjective phrase (AP).
- Adjectives can have an adverb before the adjective. Example: the least expensive fare
- A rule which defines all prenominal modifiers:

  NP → (Det) (Card) (Ord) (Quant) (AP) Nominal

After the Head Noun 1/2

- A head noun can be followed by three kinds of postmodifiers:

1. prepositional phrases – Example: all flights from Cleveland
2. non-finite clauses – Example: any flights arriving after eleven a.m.
3. relative clauses – Example: a flight that serves breakfast

- A nominal rule that accounts for PPs: Nominal → Nominal PP
- Three most common kinds of non-finite postmodifiers:

a) Gerundive (-ing)
b) -ed
c) Infinite forms

- Gerundives consist of a VP that begins with the gerundive form of the verb. Examples: any of those leaving on Thursday
- A nominal rule that accounts for gerundive modifiers: Nominal → GerundVP and the rules for gerunds are: GerundVP → GerundVP NP | GerundVP PP | GerundV | GerundV NP PP

After the Head Noun 2/2

- Postmodifiers based on –ed forms and infinities. Examples:

I need to have dinner served.

Which is the aircraft used by this flight ?

The last flight to arrive in Boston.

- Postnominal relative clauses (a.k.a restrictive relative clauses) begin with a relative pronoun (*that* or *who*). The relative pronoun functions as the subject of the embedded verb.

- Examples:

A flight that serves breakfast

Flights that leave in the morning

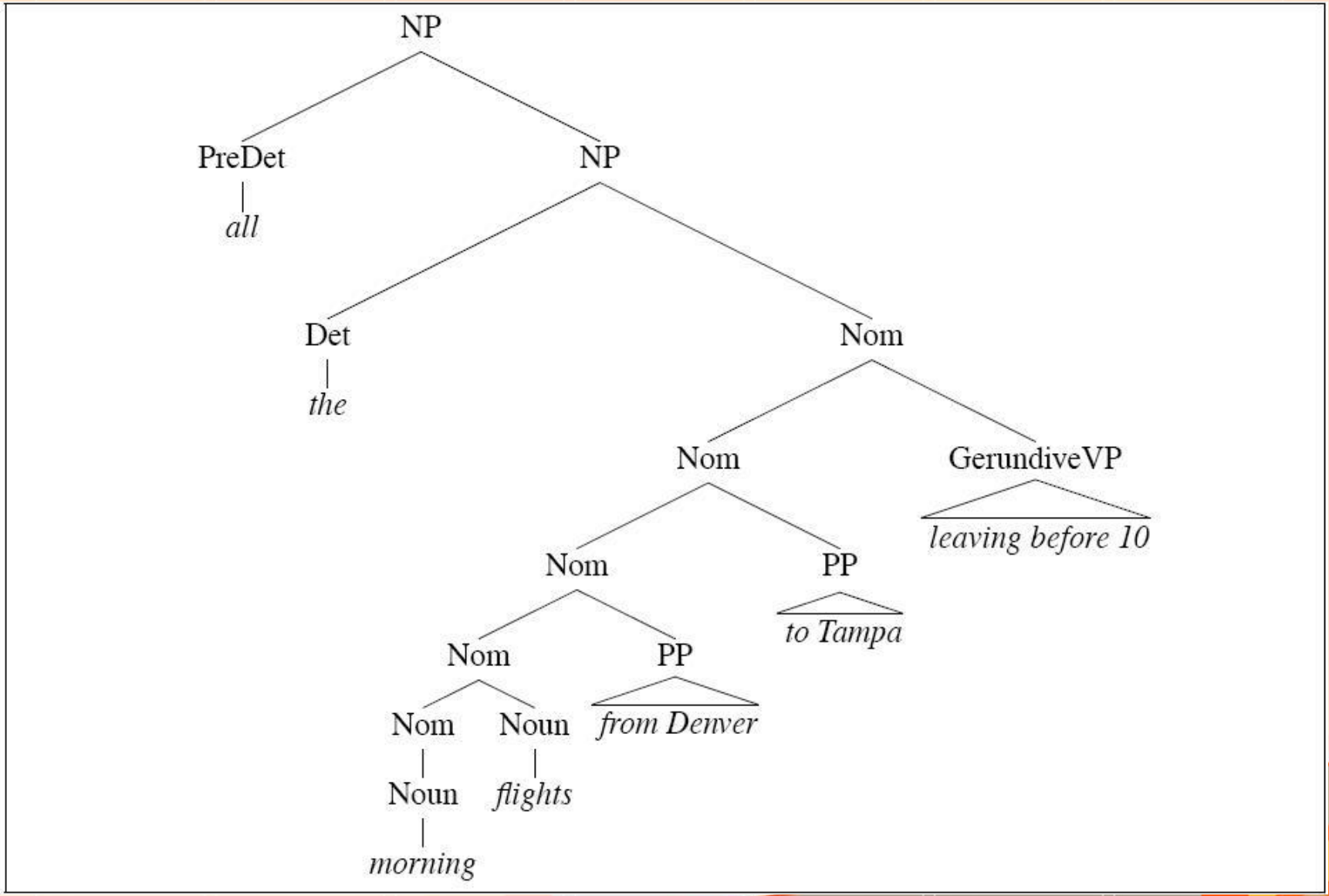
The one that leaves at ten thirty five

- To deal with relative clauses, we add the rules:

Nominal → Nominal RelClause

RelClause → (who | that) VP

Parse tree for “all the morning flights from Denver to Tampa leaving before 10”



Verb Phrase and Subcategorization

- VPs consist of a verb and a number of other constituents. These constituents include NPs and PPs:

VP → Verb Example: disappear

VP → Verb NP Example: prefer a morning flight

VP → Verb NP PP Example: leave Boston in the morning

VP → Verb PP Example: leaving on Thursday

- ❑ An entire embedded sentence may follow a verb. They are called sentential complements. Examples:

You [$_{VP}$ [$_{v}$ said [$_{S}$ there were two flights that were the cheapest]]]

[$_{VP}$ [$_{v}$ Tell] [$_{NP}$ me] [$_{s}$ how to get from the airport in Philadelphia to downtown]]

- ❑ To deal with sentential complements we add the rule:

VP → Verb S

- ❑ Another potential constituent of a VP is another VP. This happens for some verbs, e.g. want, try, would like, intend, need. Examples:

I want [$_{VP}$ to fly from Dallas to Orlando]

Hello, I'm trying [$_{VP}$ to find a flight from Dallas to Orlando]

Subcategorize

- Traditional grammars subcategorize verbs into transitive and intransitive.
- *Modern* grammars distinguish as many as 100 subcategories. The COMLEX and the AQUILEX tagsets.
- A verb like "find" subcategorizes for an NP, while a verb like "want" subcategorizes either for an NP or a non-finite VP.
- The set of possible complements are called the subcategorization frame.

Frame	Verb	Example
∅	eat, sleep	I ate
NP	prefer, find, leave	Find [$_{NP}$ the flight from Pittsburgh to Boston]
NP NP	show, give	Show [$_{NP}$ me] [$_{NP}$ airlines with flights from Pittsburgh]
PP_{from} PP_{to}	fly, travel	I would like to fly [$_{PP}$ from Boston] [$_{PP}$ to Philadelphia]
NP PP_{with}	help, load	Can you help [$_{NP}$ me] [$_{PP}$ with a flight]
VPto	prefer, want, need	I would prefer [$_{VPto}$ to go by United airlines]
VPbrst	can, would, might	I can [$_{VPbrst}$ go from Boston]
S	mean	Does this mean [$_{S}$ AA has a hub in Boston]

Coordination

- Phrases can be conjoined with conjunctions (e.g. and, or, but)
- Forms of coordinations:
- For noun phrases NP → NP and NP

Please repeat [$_{NP}$ [$_{NP}$ the flights] and [$_{NP}$ the costs]

- for nominals Nominal → Nominal and Nominal

Please repeat the [$_{Nom}$ [$_{Nom}$ flights] and [$_{Nom}$ costs]

- For verb phrases VP → VP and VP

What flights do you have [$_{VP}$ [$_{VP}$ leaving Denver] **and** [$_{Nom}$ arriving in San Francisco]

- For S conjunction

[$_{S}$ [$_{S}$ I'm interested in a flight from Denver to Dallas] **and** [$_{S}$ I'm also interested in going to San Francisco]

- Coordination makes use of metarules: X → X and X

Treebanks

- *Sufficiently robust grammars consisting of context-free grammar rules can be used to assign a parse tree to any sentence. This means that it is possible to build a corpus where every sentence in the collection is paired with a corresponding parse tree. Such a syntactically annotated corpus is called a* ***treebank.***

- Corpus in which each sentence is syntactically annotated with a parse tree

- The Penn Treebank uses corpora in English from Brown, Switchboard, ATIS and Wall Street Journal. There are treebanks for Chinese and Arabic as well.
- Other Treebanks: Prague Dependency Treebank for Czech, Negra Treebank for German, the Susanne Treebank for English.

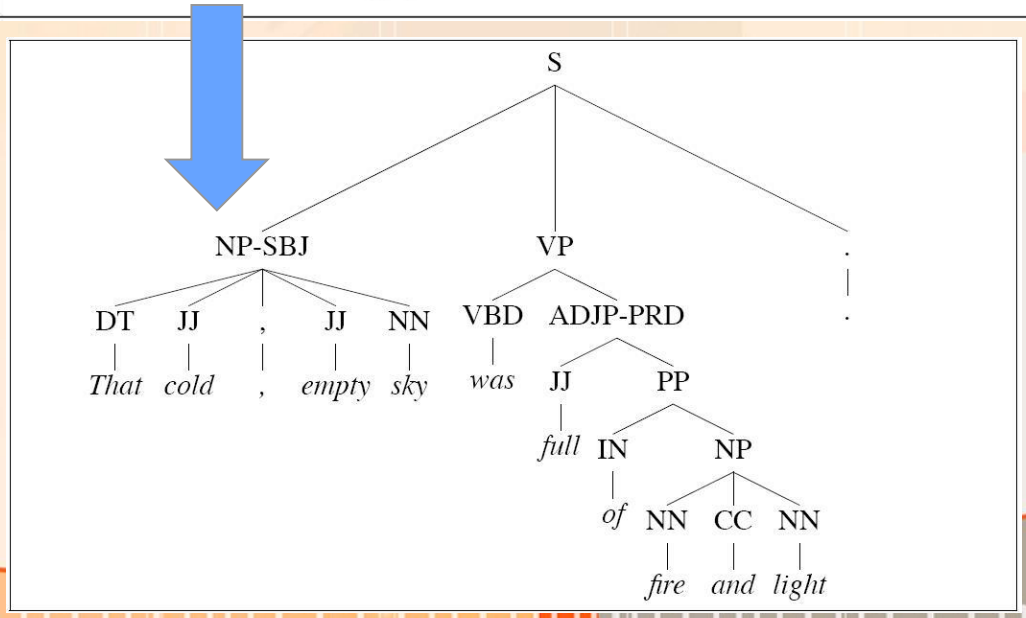
- *Universal Dependencies project (Nivre et al., 2016).*

```
((S
   (NP-SBJ (DT That)
     (JJ cold) (, ,)
     (JJ empty) (NN sky) )
   (VP (VBD was)
     (ADJP-PRD (JJ full)
       (PP (IN of)
         (NP (NN fire)
           (CC and)
           (NN light) ))))
   (. .) ))
```

(a) Brown Corpus

```
((S
   (NP-SBJ The/DT flight/NN )
   (VP should/MD
     (VP arrive/VB
       (PP-TMP at/IN
         (NP eleven/CD a.m/RB ))
       (NP-TMP tomorrow/NN )))))
```

(b) ATIS Corpus



```
( (S (`` ``)
    (S-TPC-2
      (NP-SBJ-1 (PRP We) )
      (VP (MD would)
        (VP (VB have)
          (S
            (NP-SBJ (-NONE- *-1) )
            (VP (TO to)
              (VP (VB wait)
                (SBAR-TMP (IN until)
                  (S
                    (NP-SBJ (PRP we) )
                    (VP (VBP have)
                      (VP (VBN collected)
                        (PP-CLR (IN on)
                          (NP (DT those)(NNS assets)))))))))))))
    (, ,) (`` ``)
    (NP-SBJ (PRP he) )
    (VP (VBD said)
      (S (-NONE- *T*-2) ))
    (. .) ))
```

None is used to mark Long-distance Dependencies, aka Syntactic movement. It is the complement Of the verb "said"

It is also used to mark the fact that the verb To wait has no syntactic Subject.

Using a Treebank as a Grammar

The sentences in a treebank implicitly constitute a *grammar of the language*.

For example, we can take the three parsed sentences in Figs 11.7 and Fig. 11.9 and extract each of the CFG rules in them. For simplicity, let's strip off the rule suffixes (-SBJ and so on). The resulting grammar is shown in Fig. 11.10.

```
((S
   (NP-SBJ (DT That)
     (JJ cold) (, ,)
     (JJ empty) (NN sky) )
   (VP (VBD was)
     (ADJP-PRD (JJ full)
       (PP (IN of)
         (NP (NN fire)
           (CC and)
           (NN light) ))))
   (. .) ))
```

(a) Brown Corpus

```
((S
   (NP-SBJ The/DT flight/NN )
   (VP should/MD
     (VP arrive/VB
       (PP-TMP at/IN
         (NP eleven/CD a.m/RB ))
       (NP-TMP tomorrow/NN )))))
```

(b) ATIS Corpus

```
( (S (`` ``)
    (S-TPC-2
      (NP-SBJ-1 (PRP We) )
      (VP (MD would)
        (VP (VB have)
          (S
            (NP-SBJ (-NONE- *-1) )
            (VP (TO to)
              (VP (VB wait)
                (SBAR-TMP (IN until)
                  (S
                    (NP-SBJ (PRP we) )
                    (VP (VBP have)
                      (VP (VBN collected)
                        (PP-CLR (IN on)
                          (NP (DT those)(NNS assets)))))))))))))
    (, ,) (`` ``)
    (NP-SBJ (PRP he) )
    (VP (VBD said)
      (S (-NONE- *T*-2) ))
    (. .) ))
```

Grammar	Lexicon
$S \rightarrow NP\ VP\ .$	$PRP \rightarrow we \mid he$
$S \rightarrow NP\ VP$	$DT \rightarrow the \mid that \mid those$
$S \rightarrow$ " S ", $NP\ VP\ .$	$JJ \rightarrow cold \mid empty \mid full$
$S \rightarrow$ -NONE-	$NN \rightarrow sky \mid fire \mid light \mid flight \mid tomorrow$
$NP \rightarrow DT\ NN$	$NNS \rightarrow assets$
$NP \rightarrow DT\ NNS$	$CC \rightarrow and$
$NP \rightarrow NN\ CC\ NN$	$IN \rightarrow of \mid at \mid until \mid on$
$NP \rightarrow CD\ RB$	$CD \rightarrow eleven$
$NP \rightarrow DT\ JJ\ ,\ JJ\ NN$	$RB \rightarrow a.m.$
$NP \rightarrow PRP$	$VB \rightarrow arrive \mid have \mid wait$
$NP \rightarrow$ -NONE-	$VBD \rightarrow was \mid said$
$VP \rightarrow MD\ VP$	$VBP \rightarrow have$
$VP \rightarrow VBD\ ADJP$	$VBN \rightarrow collected$
$VP \rightarrow VBD\ S$	$MD \rightarrow should \mid would$
$VP \rightarrow VBN\ PP$	$TO \rightarrow to$
$VP \rightarrow VB\ S$	
$VP \rightarrow VB\ SBAR$	
$VP \rightarrow VBP\ VP$	
$VP \rightarrow VBN\ PP$	
$VP \rightarrow TO\ VP$	
$SBAR \rightarrow IN\ S$	
$ADJP \rightarrow JJ\ PP$	
$PP \rightarrow IN\ NP$	

Using a Treebank as a Grammar

The grammar used to parse the Penn Treebank is relatively flat, resulting in very many and very long rules.

For example among the approximately 4,500 different rules for expanding VP are separate rules for PP sequences of any length, and every possible arrangement of verb arguments:

VP → VBD PP

VP → VBD PP PP

VP → VBD PP PP PP

VP → VBD PP PP PP PP

VP → VB ADVP PP

VP → VB PP ADVP

VP → ADVP VB PP[-3pt]

as well as even longer rules, such as these two:

VP → VBP PP PP PP PP PP ADVP PP

VP → VBN PP ADVP PP PP SBAR

Rules and Sentences

VP → VBP PP PP PP PP PP ADVP PP

VP → VBN PP ADVP PP PP SBAR

The two of those rules, come from the following two VPs:

This mostly happens because we [$_{VP}$ *go from football in the fall to lifting in the winter to football again in the spring*.]

Put more simply, GOP candidates for president are [$_{vp}$ *looked on more kindly by voters than Republican candidates for the Senate when the prisoner's dilemma is more severe*].

What about NPs????

Grammar rules for NP extracted from Treebanks

Some of the many thousands of NP rules include:

NP → DT JJ NN
NP → DT JJ NN NN
NP → DT JJ JJ NN
NP → DT JJ CD NNS
NP → RB DT JJ NN NN
NP → RB DT JJ JJ NNS
NP → DT JJ JJ NNP NNS
NP → DT NNP NNP NNP NNP JJ NN
NP → DT JJ NNP CC JJ JJ NN NNS
NP → RB DT JJS NN NN SBAR
NP → DT VBG JJ NNP NNP CC NNP
NP → DT JJ NNS , NNS CC NN NNS NN
NP → DT JJ JJ VBG NN NNP NNP FW NNP
NP → NP JJ , JJ “ SBAR ” NNS

Rules and Sentences

NP → DT JJ JJ VBG NN NNP NNP FW NNP

NP → NP JJ , JJ " SBAR " NNS

The two of those rules, come from the following two NPs:

NP1 =[$_{DT}$ The] [$_{JJ}$ state-owned] [$_{JJ}$ industrial] [$_{vbg}$ holding] [$_{NN}$ company] [$_{NNP}$ Instituto] [$_{NNP}$ Nacional] [$_{FW}$ de] [$_{NNP}$ Industria]

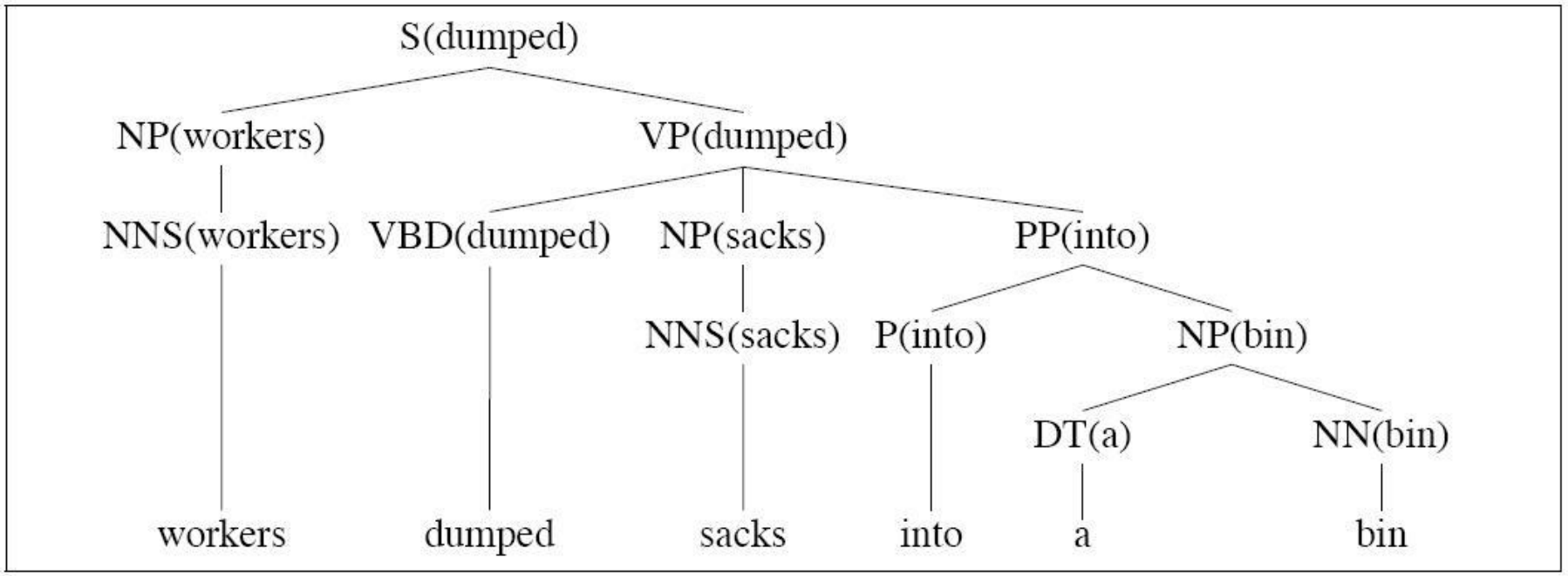
NP2 = [$_{NP}$ Shearson's] [$_{JJ}$ easy-to-film], [$_{JJ}$ black-and-white] "[$_{SBAR}$ Where We Stand]" [$_{NNS}$ commercials]

Viewed as large grammar in this way, the Penn Treebank III wall Street Journal corpus, which contains about 1 million words, also has about 1 million non-lexical rule tokens, consisting of about 17,500 distinct rule types.

Heads and Head Finding

- *Syntactic constituents could be associated with a lexical **head**; N is the head of an NP, V is the head of a VP.*
- In one simple model of lexical heads, each context-free rule is associated with a head (Charniak, 1997; Collins, 1999). The head is the word in the phrase which is grammatically the most important.

- Heads are passed up the parse tree; thus each non-terminal in a parse-tree is annotated with a single word which is its lexical head.



In order to generate such a tree, there are two possibilities:

1. each CFG rule must be augmented to identify one right-hand-side constituent to be the head daughter. The headword for a node is then set to the headword of its head daughter.

2. Instead of specifying head rules in the grammar itself, heads are identified dynamically in the context of trees for specific sentences. In other words, once a sentence is parsed, the resulting tree is walked to decorate each node with the appropriate head.

Example of rule for finding the head of an NP

If the last word is tagged POS **then** return last-word.

Else search from right to left for the first child which is an NN, NNP, NNPS, NX, POS, or JJR.

Else search from left to right for the first child which is an NP.

Else search from right to left for the first child which is a $, ADJP, or PRN.

Else search from right to left for the first child which is a CD.

Else search from right to left for the first child which is a JJ, JJS, RB or QP.

Else return the last word.

Head Percolation Table

Parent	Direction	Priority List
ADJP	Left	NNS QP NN $ ADVP JJ VBN VBG ADJP JJR NP JJS DT FW RBR RBS SBAR RB
ADVP	Right	RB RBR RBS FW ADVP TO CD JJR JJ IN NP JJS NN
PRN	Left	
PRT	Right	RP
QP	Left	$ IN NNS NN JJ RB DT CD NCD QP JJR JJS
S	Left	TO IN VP S SBAR ADJP UCP NP
SBAR	Left	WHNP WHPP WHADVP WHADJP IN DT S SQ SINV SBAR FRAG
VP	Left	TO VBD VBN MD VBZ VB VBG VBP VP ADJP NN NNS NP

Grammar Equivalence and Normal Form

- A formal language is defined as a string of words. If two grammars are equivalent, they must generate the same set of words.
- Two grammars are **strongly equivalent** if they generate the same set of strings and if they assign the same phrase structure to each sentence.
- Two grammars are **weakly equivalent** if they generate the same set of strings but they do not assign the same phrase structure to each sentence.
- It is sometimes useful to have a **normal form** for grammars, in which each of the productions take a particular form. For example, a CFG is in Chomsky Normal Form (CNF) if :
  1. It is ε-free
  2. Each production is of the form $A \rightarrow B\ C$ or $A \rightarrow a$

Binary branching

- Any grammar can be converted into a weakly-equivalent Chomsky normal form grammar.
  - Example: $A \rightarrow B\ C\ D$ can be transformed into:

$A \rightarrow B\ X$ and $X \rightarrow C\ D$

Finite-State and Context-Free Grammars

- Adequate models for grammars need to represent complex inter-related facts about constituency, sub-categorization and dependency relations.
- To accomplish this we need the power of CFGs.
- Question: Why can't we use finite-state methods to capture these phenomena?
- Answers:

1. Mathematically we can formally show that there are syntactic structures in the human languages that cannot be captured by finite-state methods.
2. Expressiveness: even when finite-state methods are capable of dealing with syntactic facts, they express them in ways in which generalizations are less obvious.

Lexicalized Grammars

- *The approach to grammar presented thus far emphasizes phrase-structure **rules** while minimizing the role of the **lexicon**.*
- *This approach leads to solutions that are cumbersome at best, yielding grammars that are redundant, hard to manage, and brittle.*
- Solutions:
  - *Lexical-Functional Grammar (LFG) (Bresnan, 1982)*
  - *Head-Driven Phrase Structure Grammar (HPSG) (Pollard and Sag, 1994)*
  - *Tree-Adjoining Grammar (TAG) (Joshi, 1985), and*
  - *Combinatory Categorial Grammar (CCG).*
- *These approaches differ with respect to how lexicalized they are — the degree to which they rely on the lexicon as opposed to phrase structure rules to capture facts about the language.*

Categorial Grammar

- Each categorial grammar has two components:

1. The **set of categories** – it associates each word with a category
2. The **combinatory rules** – *govern how categories combine in context.*

- ***Categories*** *are either atomic elements or single-argument functions that return a category as a value when provided with a desired category as argument. More formally, we can define* $\mathscr{C}$*, a set of categories for a grammar as follows:*

  - $\mathscr{A} \subseteq \mathscr{C}$, where $\mathscr{A}$ is a given set of atomic elements
  - $(X/Y), (X\backslash Y) \in \mathscr{C}$, if $X, Y \in \mathscr{C}$

The slash notation shown here is used to define the functions in the grammar. It specifies the type of the expected argument, the direction it is expected be found, and the type of the result. Thus, (X/Y) is a function that seeks a constituent of type Y to its right and returns a value of X; (X\Y) is the same except it seeks its argument to the left.

- *The set of* ***atomic categories*** *is typically very small and includes familiar elements such as sentences and noun phrases.* ***Functional categories*** *include verb phrases and complex noun phrases among others.*

Categorial Grammar

- *The slash specifies:*
  - *the type of the expected **argument,***
  - *the direction it is expected be found, and*
  - *the type of the result.*

Thus, (X/Y) is a function that seeks a constituent of type Y to its right and returns a value of X; (X\Y) is the same except it seeks its argument to the left.

- Arguments, e.g. nouns, have atomic categories – N.
- Operators: X/Y and X\Y
  - X\Y means a function f:X →Y, something that combines with Y on its right to produce an X.
  - Examples:
    - Determiners receive the category **NP\N**;
    - transitive verbs receive the category **VP\NP;**
    - ditransitive verbs receive the category (VP/NP)/NP

The lexicon

- *The lexicon in a categorial approach consists of assignments of categories to words. These assignments can either be to atomic or functional categories, and due to lexical ambiguity words can be assigned to multiple categories. Consider the following sample lexical entries.*

$$\begin{aligned} \textit{flight} &: N \\ \textit{Miami} &: NP \\ \textit{cancel} &: (S\backslash NP)/NP \end{aligned}$$

Nouns and proper nouns like flight and Miami are assigned to atomic categories, reflecting their typical role as arguments to functions.

A transitive verb like cancel is assigned the category (S\NP)/NP: a function that seeks an NP on its right and returns as its value a function with the type (S\NP). This function can, in turn, combine with an NP on the left, yielding an S as the result.

Rules

- *The rules of a categorial grammar specify how functions and their arguments combine. The following two rule templates constitute the basis for all categorial grammars:*

$$X/Y \;\; Y \;\Rightarrow\; X$$
$$Y \;\; X\backslash Y \;\Rightarrow\; X$$

The first rule applies a function to its argument on the right, while the second looks to the left for its argument. We'll refer to the first as ***forward function application****, and the second as* ***backward function application****. The result of applying either of these rules is the category specified as the value of the function being applied.*

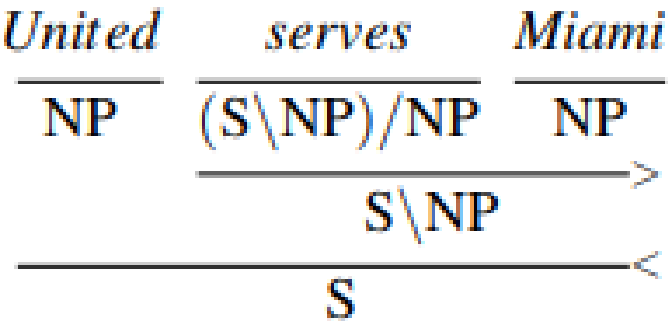
Example

Given these rules and a simple lexicon, let's consider an analysis of the sentence United serves Miami.

- *Assume that serves is a transitive verb with the category (S\NP)/NP and that United and Miami are both simple NPs. Using both forward and backward function application, the derivation would proceed as follows:*

$$X/Y \;\; Y \;\Rightarrow\; X$$

$$Y \;\; X\backslash Y \;\Rightarrow\; X$$



- *Categorial grammar derivations are illustrated growing down from the words, rule applications are illustrated with a **horizontal line** that spans the elements involved, with the type of the operation indicated at the right end of the line.*

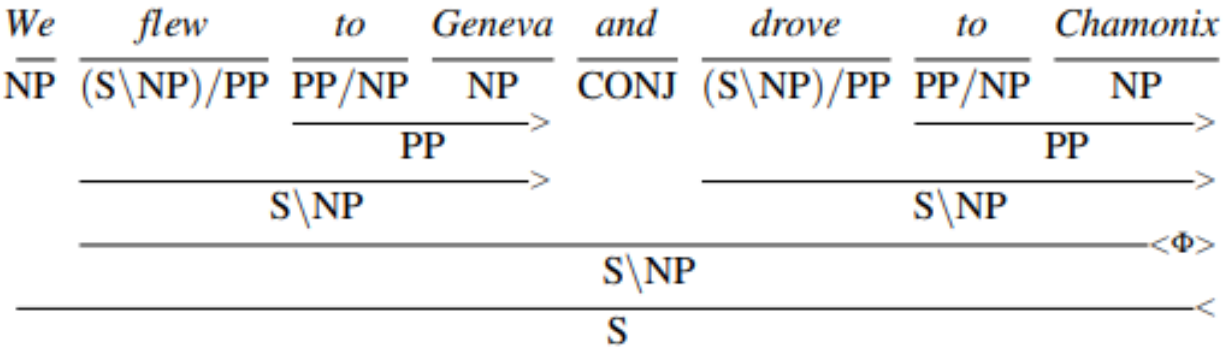
In this example, there are two function applications: one forward function application indicated by the > that applies the verb serves to the NP on its right, and one backward function application indicated by the < that applies the result of the first to the NP United on its left.

Coordination in CCG

- *Recall that English permits the coordination of two constituents of the same type, resulting in a new constituent of the same type. The following rule provides the mechanism to handle such examples:*

$$X \ CONJ \ X \Rightarrow X$$

This rule states that when two constituents of the same category are separated by a constituent of type CONJ they can be combined into a single larger constituent of the same type. The following derivation illustrates the use of this rule.



Here the two S\NP constituents are combined via the conjunction operator $< \phi >$ to form a larger constituent of the same type, which can then be combined with the subject NP via backward function application.

Operators in CCG

- *The first pair of operators permit us to compose adjacent functions.*

$$X/Y \;\; Y/Z \;\Rightarrow\; X/Z$$
$$Y\backslash Z \;\; X\backslash Y \;\Rightarrow\; X\backslash Z$$

*The first rule, called **forward composition**, can be applied to adjacent constituents where:*

- *the first is a function seeking an argument of type Y to its right, and*
- *the second is a function that provides Y as a result.*

This first rule allows us to compose these two functions into a single one with the type of the first constituent and the argument of the second.

*The second rule, backward composition is the same, except that we're looking to the left instead of backward composition to the right for the relevant arguments. Both kinds of composition are signalled by a **B** in CCG diagrams, accompanied by a < or > to indicate the direction.*

Type raising in CCG

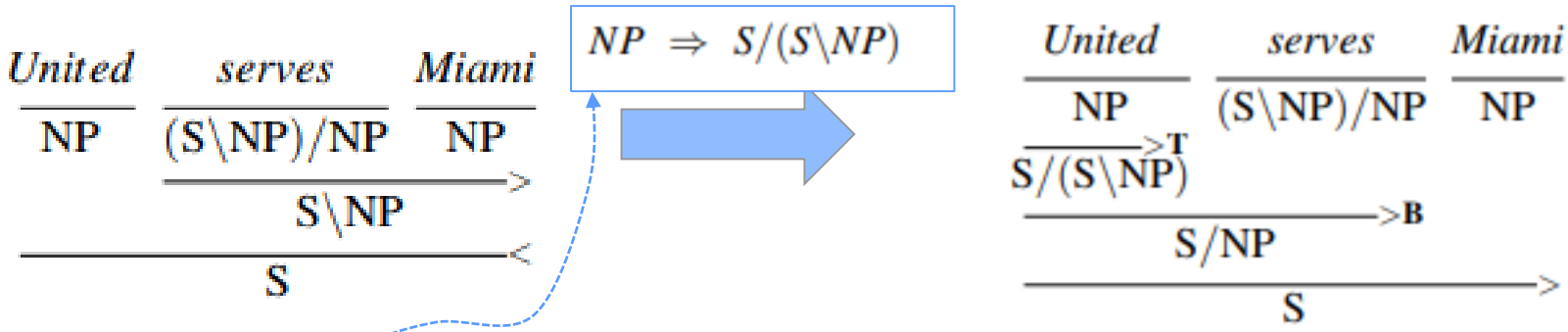
- *Type raising elevates simple categories to the status of functions.*
- *Type raising takes a category and converts it to function that seeks as an argument a function that takes the original category as its argument. The following schema show two versions of type raising: one for arguments to the right, and one for the left.*

$$X \Rightarrow T/(T\backslash X)$$
$$X \Rightarrow T\backslash(T/X)$$

The category T in these rules can correspond to any of the atomic or functional categories already present in the CCG grammar.

Example of Type raising in CCG

- *A particularly useful example of type raising transforms a simple NP argument in subject position to a function that can compose with a following VP.*
- *Let's revisit our earlier example of United serves Miami.*
  - *Instead of classifying United as an NP which can serve as an argument to the function attached to serve, we can use type raising to reinvent it as a function in its own right as follows.*



Combining this type-raised constituent with the forward composition rule permits the following alternative to the previous derivation.

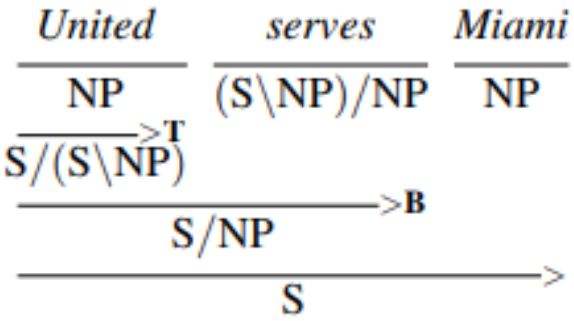
By type raising United to S/(S\NP), we can compose it with the transitive verb serves to yield the (S/NP) function needed to complete the derivation.

$$X \Rightarrow T/(T\backslash X)$$
$$X \Rightarrow T\backslash(T/X)$$

More on the Example of Type raising in CCG

- *Some interesting things to note about this derivation:*

1. *it provides a left-to-right, word-by-word derivation that more closely mirrors the way humans process language. This makes CCG a particularly apt framework for psycholinguistic studies.*
2. *This derivation involves the use of an intermediate unit of analysis, United serves, that does not correspond to a traditional constituent in English. Let's revisit our earlier example of United serves Miami.*



More examples:

- Consider the sentence from Mark Steedman:

  Harry eats apples.

- Let us assume that the VP="eats apples" has the category (S\NP).
- The parse of the verb in the VP is (S\NP)/NP because both Harry and apples are NPs.
- The parse of the sentence"

Harry	*eats*	*apples.*
NP	(S\NP)/NP	NP
	S\NP	
	S	

Summary

- *We introduced a number of fundamental concepts in syntax through the use of context-free grammars.*

• *In many languages, groups of consecutive words act as a group or a constituent, which can be modeled by context-free grammars.*

• *A context-free grammar consists of a set of rules or productions, expressed over a set of* ***non-terminal symbols*** *and a* ***set of terminal symbols****.*

• *There are many sentence-level grammatical constructions in English; declarative, imperative, yes-no question, and wh-question are four common types; these can be modeled with context-free rules.*

• *An English noun phrase can have determiners, numbers, quantifiers, and adjective phrases preceding the head noun, which can be followed by a number of postmodifiers; gerundive VPs, infinitives VPs, and past participial VPs are common possibilities.*

• *Subjects in English agree with the main verb in person and number.*

• *Verbs can be subcategorized by the types of complements they expect. Simple subcategories are transitive and intransitive; most grammars include many more categories than these.*

• *Treebanks of parsed sentences exist for many genres of English/other languages.*

• ***Combinatorial categorial grammar*** *(CCG) is an important computationally relevant lexicalized approach.*