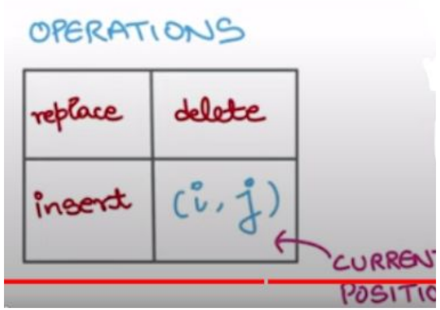
# This is AI4001

GCR : t37g47w

# Minimum Edit Distance

It measures the minimum number of single-character edits (insertions, deletions, or substitutions) required to transform one string into another.



# Minimum Edit Distance

The operations allowed in minimum edit distance calculations are typically defined as follows:

- Insertion: Adding a character to one of the strings.
- Deletion: Removing a character from one of the strings.
- Substitution: Replacing one character with another in one of the strings.

The goal is to find the sequence of these operations that transforms one string into the other while minimizing the total number of operations.

# Minimum Edit DIstance

```
I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N
d s s   i s
```

**Figure 2.13** Representing the minimum edit distance between two strings as an **alignment**. The final row gives the operation list for converting the top string into the bottom string: d for deletion, s for substitution, i for insertion.

```
function MIN-EDIT-DISTANCE(source, target) returns min-distance

  n ← LENGTH(source)
  m ← LENGTH(target)
  Create a distance matrix distance[n+1,m+1]

  # Initialization: the zeroth row and column is the distance from the empty string
      D[0,0] = 0
      for each row i from 1 to n do
         D[i,0] ← D[i-1,0] + del-cost(source[i])
      for each column j from 1 to m do
         D[0,j] ← D[0, j-1] + ins-cost(target[j])

  # Recurrence relation:
  for each row i from 1 to n do
      for each column j from 1 to m do
         D[i, j] ← MIN( D[i−1, j] + del-cost(source[i]),
                        D[i−1, j−1] + sub-cost(source[i], target[j]),
                        D[i, j−1] + ins-cost(target[j]))
  # Termination
  return D[n,m]
```

**Figure 2.16** The minimum edit distance algorithm, an example of the class of dynamic programming algorithms. The various costs can either be fixed (e.g., $\forall x, \text{ins-cost}(x) = 1$) or can be specific to the letter (to model the fact that some letters are more likely to be inserted than others). We assume that there is no cost for substituting a letter for itself (i.e., $\text{sub-cost}(x,x) = 0$).

| Src\Tar | **#** | **e** | **x** | **e** | **c** | **u** | **t** | **i** | **o** | **n** |
|---|---|---|---|---|---|---|---|---|---|---|
| **#** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| **i** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 6 | 7 | 8 |
| **n** | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 7 | 8 | 7 |
| **t** | 3 | 4 | 5 | 6 | 7 | 8 | 7 | 8 | 9 | 8 |
| **e** | 4 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 9 |
| **n** | 5 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 10 |
| **t** | 6 | 5 | 6 | 7 | 8 | 9 | 8 | 9 | 10 | 11 |
| **i** | 7 | 6 | 7 | 8 | 9 | 10 | 9 | 8 | 9 | 10 |
| **o** | 8 | 7 | 8 | 9 | 10 | 11 | 10 | 9 | 8 | 9 |
| **n** | 9 | 8 | 9 | 10 | 11 | 12 | 11 | 10 | 9 | 8 |

**Figure 2.17** Computation of minimum edit distance between *intention* and *execution* with the algorithm of Fig. 2.16, using Levenshtein distance with cost of 1 for insertions or deletions, 2 for substitutions.

# Minimum Edit Distance

Transform “money” to “monkey”

Generate a distance matrix too!

# Applications

Spell checking and correction

Text similarity and plagiarism detection

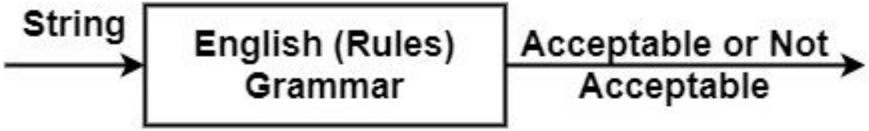
Machine translation

Speech recognition

# Formal Grammar

# Grammar

Grammar - It is a set of rules which checks whether a string belongs to a particular language or not.




Grammar

A program consists of various strings of characters. But, every string is not a proper or meaningful string. So, to identify valid strings, some rules should be specified. These rules are nothing but Grammar.

# Regular Languages

Examples: Examples of regular languages include languages that describe simple patterns like regular expressions for matching email addresses, URLs, or numeric literals in programming languages.

Applications: Regular languages are commonly used in lexical analysis (tokenization) for compilers, in text searching and pattern matching, and in many simple string-processing tasks.

# Context Free Languages

Examples: Examples of context-free languages include programming languages (which have complex syntax and nested structures), HTML or XML documents (which have hierarchical structures), and many natural language constructs.

Applications: Context-free languages play a crucial role in parsing and analyzing the syntax of programming languages, markup languages, and natural languages. They are used in compilers, parsers, syntax checkers, and other language processing tools.

# Context Free Grammar

# Context Free Grammar

It is a notation used to specify the syntax of the language. Context-free Grammar is used to design parsers.

Lexical Analyzer generates a string of tokens which are given to the parser to construct a parse tree. But, before constructing the parse tree, these tokens will be grouped so that the results of grouping will be a valid construct of a language. So, to specify constructs of language, a suitable notation is used, which will be precise & easy to understand. This notation is Context-Free Grammar.

she saw the city

| | | | |
|---|---|---|---|
| N | V | D | N |
| she | saw | the | city |

# Basic Terms

Terminals: These are the characters that make up the actual content of the final sentence. These can include words or letters depending on which of these is used as the basic building block of a sentence.

Non Terminals: These are also called variables. These act as a sub language within the language defined by the grammar. Non terminals are placeholders for the terminals. We can use non terminals to generate different patterns of terminal symbols.

Start Symbol: a start symbol is a special non terminal that represents the initial string that will be generated by the grammar.

N → she | city | car | Harry | ...

D → the | a | an | ...

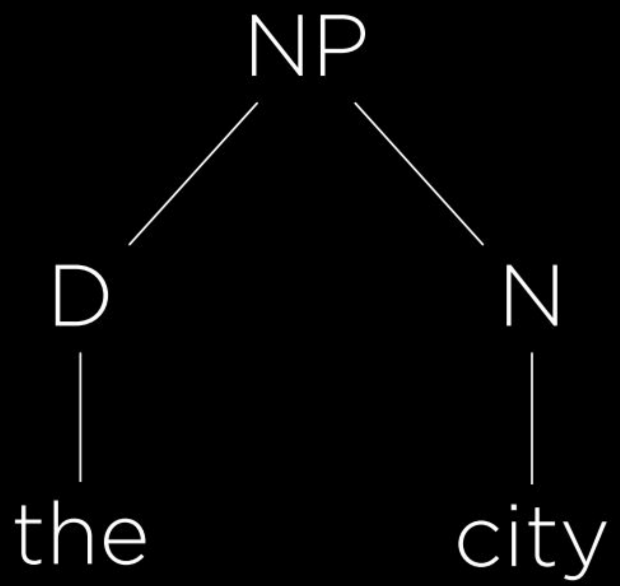
V → saw | ate | walked | ...

P → to | on | over | ...

ADJ → blue | busy | old | ...

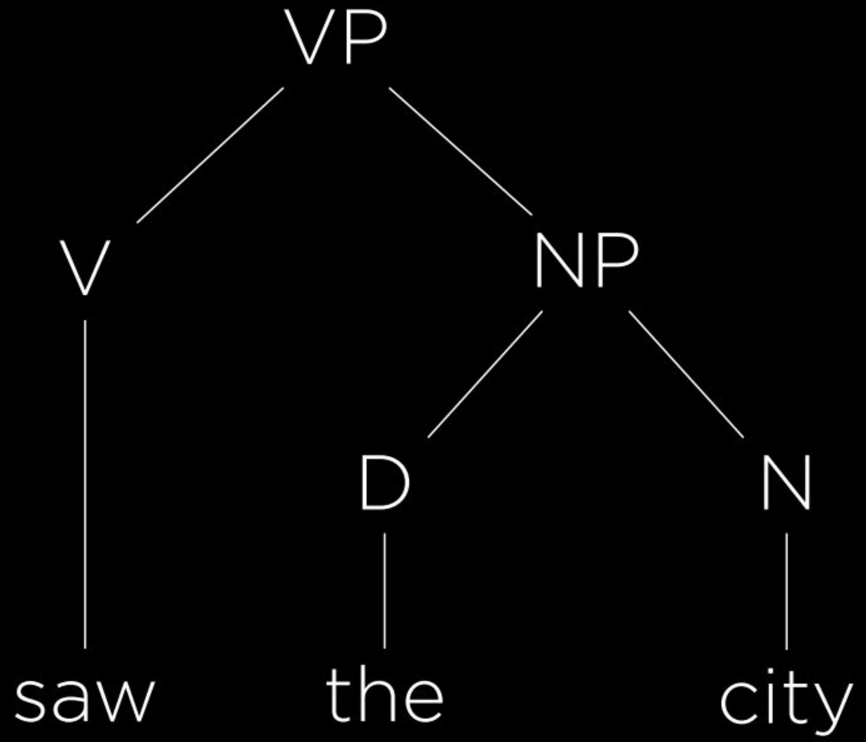
$$NP \rightarrow N \mid D\ N$$

NP → N | D N



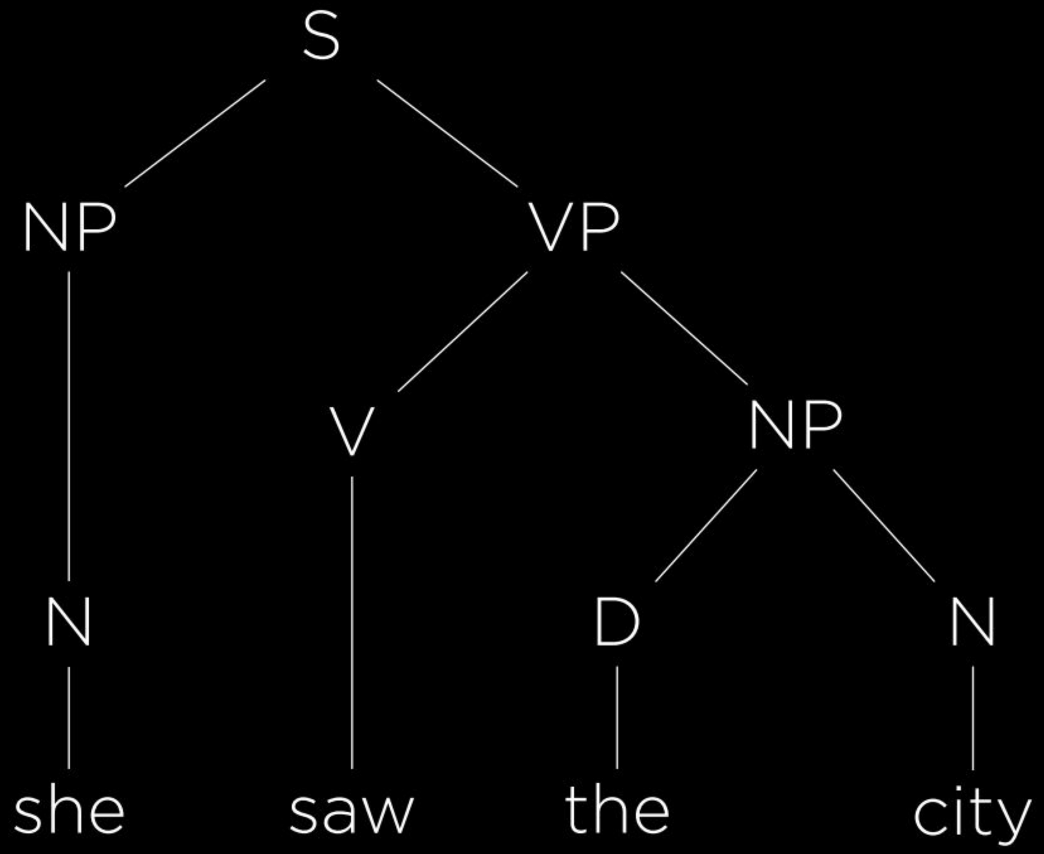
$$\mathrm{VP} \rightarrow \mathrm{V} \;\middle|\; \mathrm{V}\ \mathrm{NP}$$

$$VP \rightarrow V \mid V\ NP$$



$$S \rightarrow NP \ VP$$

$S \rightarrow NP\ VP$



# CFG

It's called "context-free" because the rules for forming valid sentences in the language are based solely on the syntax of the language, not on the context in which the sentence appears.

They are used in compilers, natural language processing, and more.

CFGs are context-free because they rely on the syntax alone, regardless of context.

Also called phase structured grammar

# Basic Terms

A context-free grammar G is defined by four parameters:

N, Σ, R, S (technically this is a “4-tuple”).

| | |
|---|---|
| $N$ | a set of **non-terminal symbols** (or **variables**) |
| $\Sigma$ | a set of **terminal symbols** (disjoint from $N$) |
| $R$ | a set of **rules** or productions, each of the form $A \rightarrow \beta$ , |
| | where $A$ is a non-terminal, |
| | $\beta$ is a string of symbols from the infinite set of strings $(\Sigma \cup N)*$ |
| $S$ | a designated **start symbol** and a member of $N$ |

# Lexicon

A lexicon refers to a collection or repository of words, phrases, or symbols in a language along with their associated information.

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

**Figure 10.3** The grammar for $\mathscr{L}_0$, with example phrases for each rule.

# Determiners

Determiners are words that come before nouns to specify or determine which noun is being referred to and to provide additional information about the noun.

the, a , an, this, that, these, my, your, his, all, many

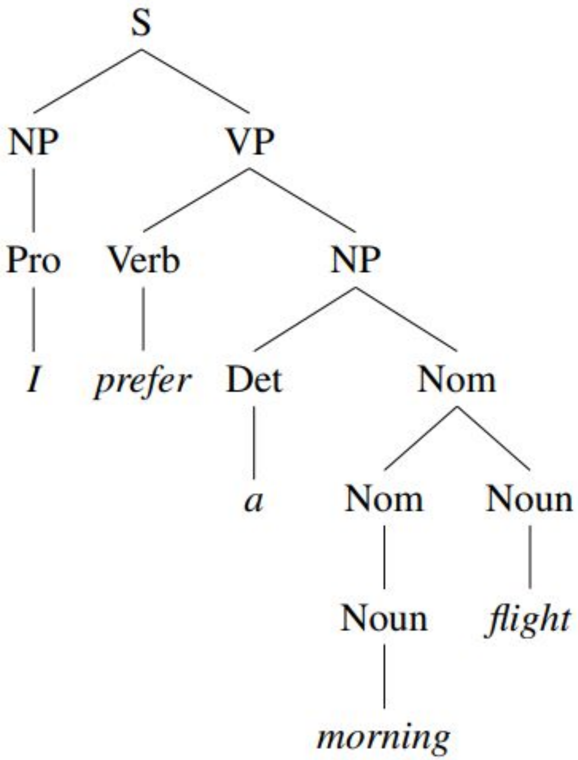
# Context Free Grammar

| | |
|---|---|
| *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* |

**Figure 10.2** The lexicon for $\mathscr{L}_0$.

# "I prefer a morning flight"

# "I prefer a morning flight"



N = {S, < NP >, < VP >, < AdjP >,   < N>, < V>, < Adj>}

Start variable = S

Σ = {big, stout, Kiao, bought, white, car, Henry, cheese, ate, green}

Rules:

S -> NP VP

NP -> N |AdjP N

VP -> V NP

AdjP -> Adj

N -> Kiao | car | Henry | cheese

V -> bought | ate

Adj -> big | stout | white | green

# Sentence Level Constructions

There are Four particularly common and important:

- Declaratives
- Imperatives
- Yes-no questions
- Wh-questions

# Declarative Sentences

Declarative structure have a subject noun phrase followed by a verb phrase

I prefer a morning flight.

I want a flight from Ontario to Chicago.

The flight should be eleven a.m. tomorrow.

The return flight should leave at around seven p.m.

# Imperative Sentences

They often begin with a verb phrase and have no subject.

They are called imperative because they are almost always used for commands and suggestions.

- Show the lowest fare.
- Give me Sunday’s flights arriving in Las Vegas from New York City.
- List all flights between five and seven p.m.

$$S \rightarrow VP$$

# Yes No Questions Sentences

Sentences with yes-no question structure are often (though not always) used to ask questions; such as asking, requesting, or suggesting.

They begin with an auxiliary verb, followed by a subject NP, followed by a VP.

Third example is not a question at all but a request;

- Do any of these flights have stops?
- Does American's flight eighteen twenty five serve dinner?
- Can you give me the same information for United?

S → Aux NP VP

# WH-phrase Sentences

The most complex sentence-level structures are the various wh structures. (who, whose, when, where, what, which, how, why).

These may be broadly grouped into two classes of sentence-level structures.

- wh-subject-question structure
- wh-non-subject-question structure

# WH-SUBJECT-QUESTION SENTENCES

The wh-subject-question structure is identical to the declarative structure, except that the first noun phrase contains some wh-word.

- What airlines fly from Burbank to Denver?
- Which flights depart Burbank after noon?
- Whose flights serve breakfast?

$$S \rightarrow \text{Wh-NP VP}$$

# Wh-non-subject-question Sentences

In the wh-non-subject-question structure, the wh-phrase is not the subject of sentence, and so the sentence includes another subject.

In these types of sentences the auxiliary appears before the subject NP, just as in the yes-no question structures.

- What flights do you have from Burbank to Tacoma Washington?

S → Wh-NP Aux NP VP

| Grammar | Lexicon |
|---|---|
| $S \rightarrow NP\ VP$ | $Det \rightarrow that \mid this \mid the \mid a$ |
| $S \rightarrow Aux\ NP\ VP$ | $Noun \rightarrow book \mid flight \mid meal \mid money$ |
| $S \rightarrow VP$ | $Verb \rightarrow book \mid include \mid prefer$ |
| $NP \rightarrow Pronoun$ | $Pronoun \rightarrow I \mid she \mid me$ |
| $NP \rightarrow Proper\text{-}Noun$ | $Proper\text{-}Noun \rightarrow Houston \mid NWA$ |
| $NP \rightarrow Det\ Nominal$ | $Aux \rightarrow does$ |
| $Nominal \rightarrow Noun$ | $Preposition \rightarrow from \mid to \mid on \mid near \mid through$ |
| $Nominal \rightarrow Nominal\ Noun$ | |
| $Nominal \rightarrow Nominal\ PP$ | |
| $VP \rightarrow Verb$ | |
| $VP \rightarrow Verb\ NP$ | |
| $VP \rightarrow Verb\ NP\ PP$ | |
| $VP \rightarrow Verb\ PP$ | |
| $VP \rightarrow VP\ PP$ | |
| $PP \rightarrow Preposition\ NP$ | |

**Figure 11.1** The $\mathscr{L}_1$ miniature English grammar and lexicon.

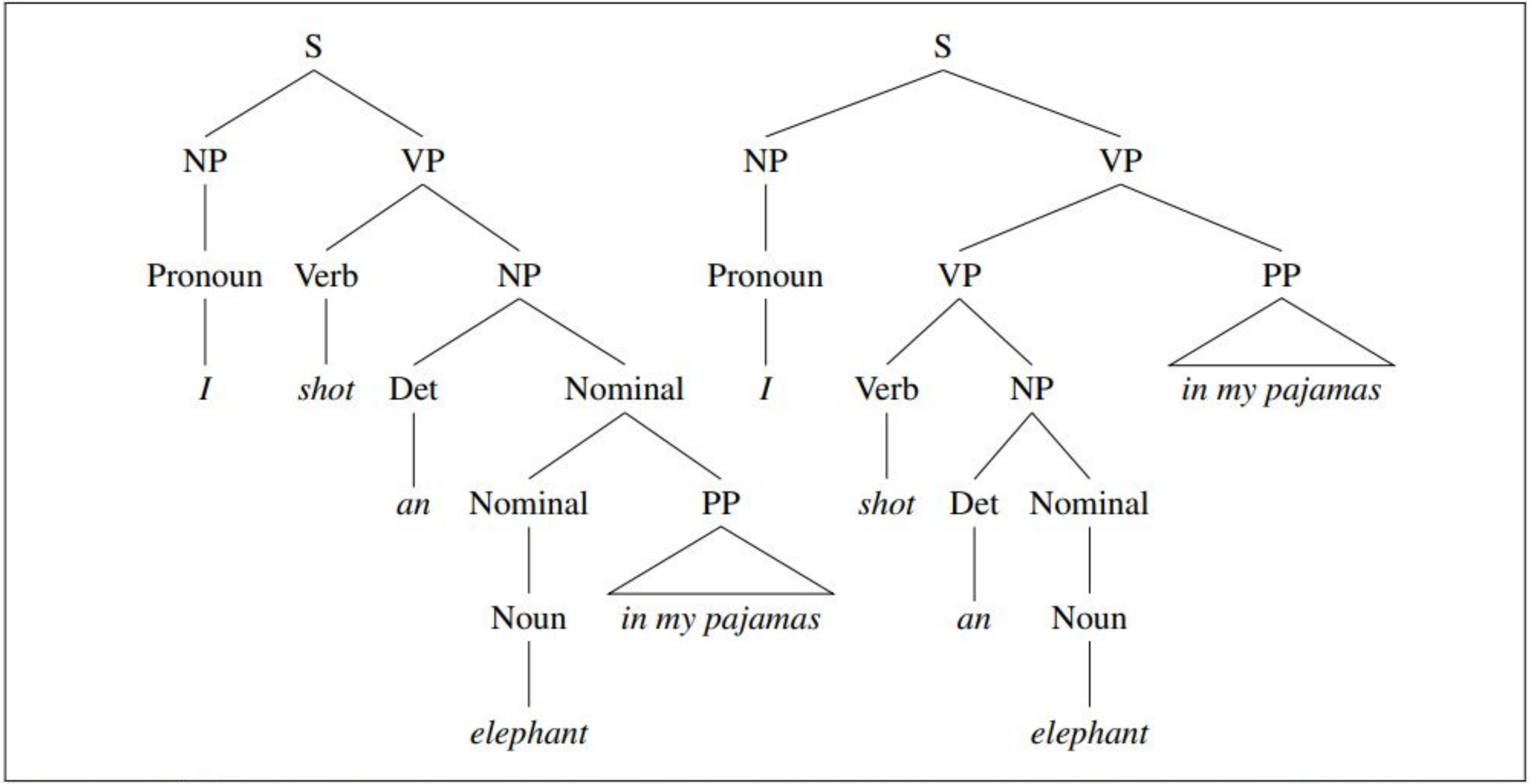



**Figure 11.2** Two parse trees for an ambiguous sentence. The parse on the left corresponds to the humorous reading in which the elephant is in the pajamas, the parse on the right corresponds to the reading in which Captain Spaulding did the shooting in his pajamas.

# Probabilistic Context Free Grammar

Also known as the Stochastic Context-Free Grammar

| | |
|---|---|
| $N$ | a set of **non-terminal symbols** (or **variables**) |
| $\Sigma$ | a set of **terminal symbols** (disjoint from $N$) |
| $R$ | a set of **rules** or productions, each of the form $A \rightarrow \beta\ [p]$,<br>where $A$ is a non-terminal,<br>$\beta$ is a string of symbols from the infinite set of strings $(\Sigma \cup N)*$,<br>and $p$ is a number between 0 and 1 expressing $P(\beta \mid A)$ |
| $S$ | a designated **start symbol** |

| 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] | \| $meal$ [.015] \| $money$ [.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] | |

**Figure 12.1** A PCFG that is a probabilistic augmentation of the $\mathscr{L}_1$ miniature English CFG grammar and lexicon of Fig. 11.1. These probabilities were made up for pedagogical purposes and are not based on a corpus (since any real corpus would have many more rules, so the true probabilities of each rule would be much smaller).

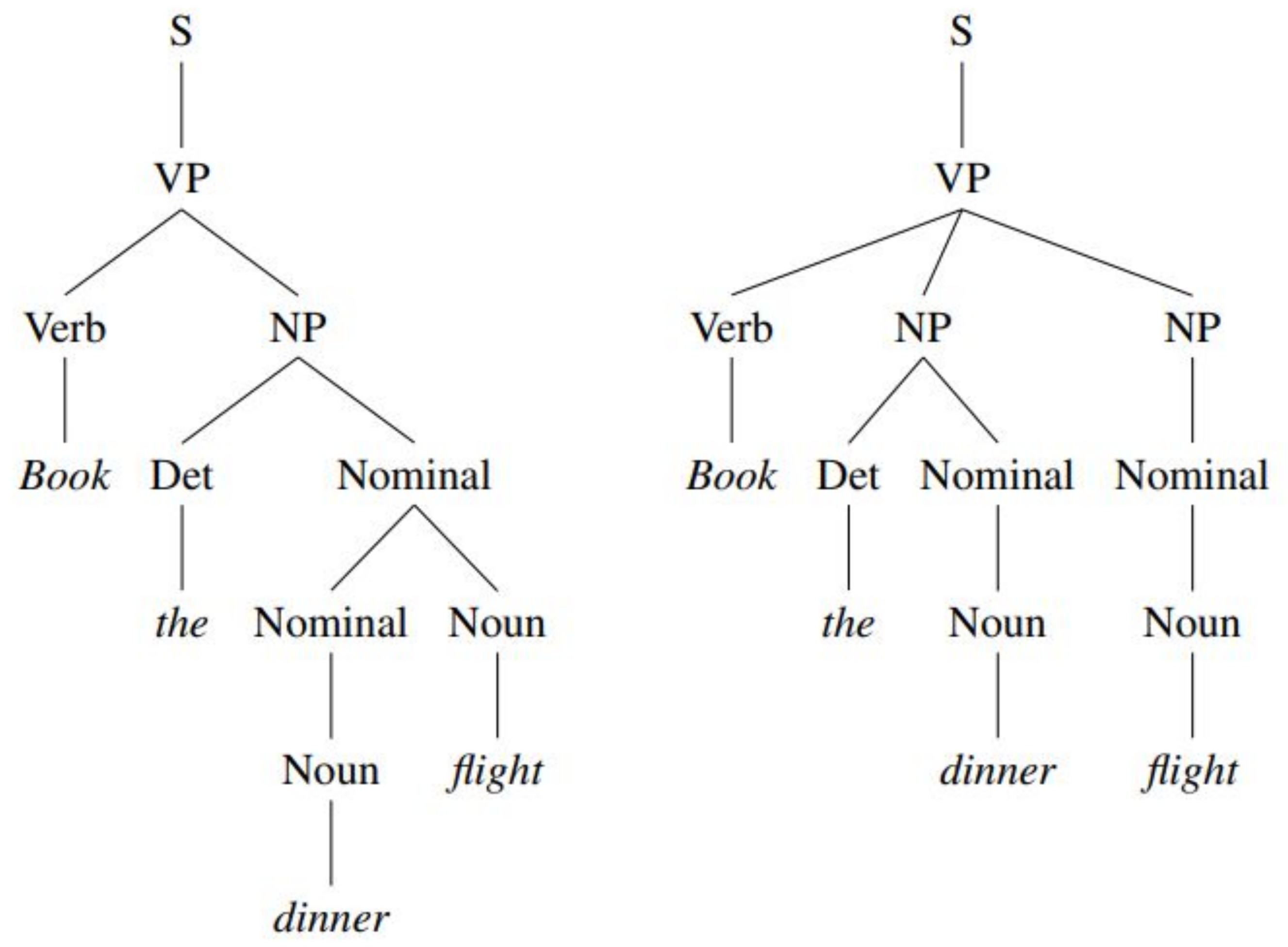
# PCFG

“Book the dinner flight”

The sensible parse means “Book a flight that serves dinner”.

The nonsensical parse mean something like “Book a flight on behalf of ‘the dinner”’

just as a structurally similar sentence like “Can you book John a flight?” means something like “Can you book a flight on behalf of John?”


S
VP
Verb
NP
Book
Det
Nominal
the
Nominal
Noun
Noun
flight
dinner
S
VP
Verb
NP
NP
Book
Det
Nominal
Nominal
the
Noun
Noun
dinner
flight

# PCFG

The probability of a particular parse T is defined as the product of the probabilities of all the n rules used to expand each of the n non-terminal nodes in the parse tree T, where each rule i can be expressed as LHSi → RHSi:

$$P(T,S) = \prod_{i=1}^{n} P(RHS_i | LHS_i)$$

$$P(T,S) = P(T)P(S|T)$$

| Rules | | | P | Rules | | | P |
|---|---|---|---|---|---|---|---|
| S | $\rightarrow$ | VP | .05 | S | $\rightarrow$ | VP | .05 |
| VP | $\rightarrow$ | Verb NP | .20 | VP | $\rightarrow$ | Verb NP NP | .10 |
| NP | $\rightarrow$ | Det Nominal | .20 | NP | $\rightarrow$ | Det Nominal | .20 |
| Nominal | $\rightarrow$ | Nominal Noun | .20 | NP | $\rightarrow$ | Nominal | .15 |
| Nominal | $\rightarrow$ | Noun | .75 | Nominal | $\rightarrow$ | Noun | .75 |
| | | | | Nominal | $\rightarrow$ | Noun | .75 |
| Verb | $\rightarrow$ | book | .30 | Verb | $\rightarrow$ | book | .30 |
| Det | $\rightarrow$ | the | .60 | Det | $\rightarrow$ | the | .60 |
| Noun | $\rightarrow$ | dinner | .10 | Noun | $\rightarrow$ | dinner | .10 |
| Noun | $\rightarrow$ | flight | .40 | Noun | $\rightarrow$ | flight | .40 |

**Figure 12.2** Two parse trees for an ambiguous sentence. The parse on the left corresponds to the sensible meaning "Book a flight that serves dinner", while the parse on the right corresponds to the nonsensical meaning "Book a flight on behalf of 'the dinner' ".

$$
\begin{aligned}
P(T_{left}) &= .05 * .20 * .20 * .20 * .75 * .30 * .60 * .10 * .40 = \mathbf{2.2 \times 10^{-6}} \\
P(T_{right}) &= .05 * .10 * .20 * .15 * .75 * .75 * .30 * .60 * .10 * .40 = \mathbf{6.1 \times 10^{-7}}
\end{aligned}
$$

# Tasks

https://scholar.harvard.edu/files/harrylewis/files/section5_sols.pdf

https://scholar.harvard.edu/files/harrylewis/files/section4_sols.pdf

https://timhagmann.com/html/e63/hw8-hagmann-tim.html

https://coli-saar.github.io/cl20/notebooks/CFGs.html

# References


Language - Lecture 6 - CS50's Introduction to Artificial Intelligence with Python

https://www.youtube.com/watch?app=desktop&v=Dd_NgYVOdLk

https://people.cs.umass.edu/~mccallum/courses/inlp2007/lect5-cfg.pdf