rules such as $Np$.

Matters become more complicated when we consider recursive

defined by $D$, and the denoted by $N$. The concatenation of the
language denoted by $N$ contains the language denoted by $p$.
For a more complex rule such as $Np$.

The non-terminal includes the alphabet.

A rule such as $N$.

Each non-terminal symbol in a grammar denotes a language.

Context-Free Grammars: Language

For a rule $A \rightarrow \alpha$, $A$ is the rule's head and $\alpha$ is its body.

- $A \in \sum \cup \sum^*$
- $\alpha \in \sum^*$
- $\sum$ is the start symbol.
- $S \in \{\sum \cup \sum^*\}$
- $A \cup \{A\}$ is a finite, non-empty set of grammar variables.
- $\sum$ is a finite, non-empty set of terminals.
- $\alpha$ is a finite, non-empty set of terminals, the alphabet.

A context-free grammar (CFG) is a four-tuple $(\sum, \sum^*, S, \rho)$.

The rules:

The start symbol is $Np$

$$\{p, Np, \rho \} = \sum$$

$$\{p, \text{the, car, in, that}\} = \sum$$
Since the form \( \langle \text{the car} \rangle \) consists of terminal symbols only, no non-terminal can be selected and hence it derives no form.

Given the form \( \langle \text{the N} \rangle \), there is exactly one non-terminal that can be selected, namely \( D \). As there exists a grammar rule whose head is \( D \) and whose body is empty, the left context is again empty, while the right context is \( \langle \text{the N} \rangle \). Hence we can apply the same process to \( \langle \text{the N} D \rangle \).

**Definition:** example

\( \langle \text{the N} D \rangle \leftarrow \langle \text{N} D \rangle \). Therefore, \( \langle \text{N} D \rangle \leftarrow \langle \text{N} \rangle \). Observe that it can be written as \( \langle \text{N} \rangle \leftarrow \langle \text{N} D \rangle \). Therefore, the form is a good candidate for derivation.

The set of non-terminals that hold between two forms, each a sequence of grammar symbols, is a relation that holds between two forms, each a non-terminal and the set of all sequences of terminal and non-terminal symbols.

Given a grammar \( \mathcal{G} \), we define the set of forms derivable from it as:

\[
\mathcal{L}(\mathcal{G}) = \{ \langle \text{S} \rangle \}.
\]
be trans-context-free.

A language that can be generated by some CFG is a context-

\[ L(G) = \{ m \in \mathcal{A} \mid m \} \]

The language generated by the grammar is the language of every member of which can be generated by a grammar with the start symbol.

A form is a sentential form of a grammar if it is generated by a grammar with the start symbol.

\[ \langle \text{the cat} \rangle \rightarrow \langle \text{NP} \rangle \]

\[ \langle \text{NP} \rangle \leftarrow \langle \text{the cat} \rangle \]

\[ \langle \text{the cat} \rangle \leftarrow \langle \text{NP} \rangle \]

\[ \langle \text{NP} \rangle \leftarrow \langle \text{the cat} \rangle \]

\[ \langle \text{NP} \rangle \leftarrow \langle \text{NP} \rangle \]

For every \( i \geq 1 \) and \( n \neq 0 \), the \( i \)-th \( \varphi \) is an initial form of \( \varphi \).

A \( \varphi \)-derivation is a sequence of \( \varphi \)-forms \( \varphi_1, \varphi_2, \ldots, \varphi_n \) such that for every \( i \geq 1 \) and \( n \neq 0 \), the \( i \)-th \( \varphi \) is a \( \varphi \)-derivation of \( \varphi \) in \( n \) steps.
namely the rule $S \rightarrow A^0 \ V^0$ is in itself recursive.

In $G^0$, the recursion is simple: the chain of rules is of length $0$.

The head of $p^1$ occurs in the body of $p^2$.

The head of $p^1$ occurs in the body of $p^2$ and the

$1 < |u| \leq |v|$. The head of $p^1$ occurs in the body of $p^2$.

exists a chain of rules $p^1, p^2, \ldots, p^n$, such that for every

Put formally, a grammar $G^0$ is recursive if there

the head of the rule can be derived.

at least one rule whose body contains a symbol, from which

To be able to produce infinitely many words with a finite

The language $L(G^0)$ is a finite grammar.

The language $L(G^0)$ is infinite: it includes an infinite number

Recursion

Language: a formal example $G^0$

\[
\{0 < u \mid u \in \mathcal{Q}_u S \} = (\mathcal{Q}T)
\]

\[
\begin{align*}
q &\leftarrow \varepsilon \\
\varphi &\leftarrow \varepsilon \\
\epsilon &\leftarrow S \\
\varepsilon &\leftarrow S
\end{align*}
\]

Similarly, $L(N)I$ and $\{u \} = (\mathcal{Q}T)$

It is fairly easy to see that $L(D)$

\[
\begin{align*}
\epsilon &\leftarrow p \\
\epsilon &\leftarrow p \\
\epsilon &\leftarrow p
\end{align*}
\]

For the example grammar (with $\epsilon$ the start symbol):

Language of a grammar

Language of a grammar

Language of a grammar

Language of a grammar
them as sisters. daughter of child of a. If n has two daughters, we refer to

d this n. Then we say that a is the mother of n and n is the

d vertex of a branch leading to which leads to some

of a vertex. A has a branch leading to which leads to some

When talking about trees, we sometimes use family notation:

(branches).

other vertex is accessible from it (by following one or more

has two properties: Its is not the target of any arc, and every

In addition, a tree has a distinguished vertex, the root, which

vertices.

set of branches (or arcs), each of which is an ordered pair of

formally, a tree consists of a finite set of vertices and a finite

Derivation tree

An example for a derivation tree for the string the cat in the

Derivation tree: example

Trees is introduced. To this end, the notion of derivation

string, it is sometimes useful to collapse such equivalent

Since both derivations use the same rules to derive the same

the cat in two ways:

Starting with the form \( \langle dN \rangle \), it is possible to derive the string

only in the order in which they were applied. Sometimes derivations

sometimes derivations provide more information than is

Derivation tree
Correspondence between trees and derivations

Derivation trees

1. A non-terminal symbol is the start symbol.
2. Every vertex has a label, which is either a terminal symbol.
3. If a vertex has an outgoing branch, its label must be a non-terminal symbol.
4. If a vertex is labeled c, it is the only child of its mother.
The label of the root is the start symbol.

The frontier, or yield of the tree, when read from left to right, the sequence of leaves is called a tree induces a natural “left-to-right” order on its leaves.

A leaf is a vertex with no outgoing branches.

The car in the hat

is a child, must be labeled by the body of the same rule.

Each non-leaf vertex in the tree corresponds to some grammar from each other only in the order in which rules are applied.

Correspondence between trees and derivations

Sometimes there exist different derivations of the same sentence that correspond to a single tree. In fact, the tree structure yields a single, possibly non-unique, representation to collapse exactly those derivations that differ.

For a form a, a non-terminal symbol A denotes a if and only if a is the yield of some parse tree whose root is A. Derivation trees correspond very closely to derivations.
Corresponding to the two readings, are available for this string:

( the car in the hat ) in the hat

( the car in the hat ) in the hat

which a certain cat-in-a-hat is inside a hat:

one in which a certain cat wears a hat-in-a-hat, and one in:

intuitively there can be (at least) two readings for this string:

the cat in the hat in the hat

Consider again the example grammar and the following string:

Correspondence between trees and derivations

Ambiguity

example: example

Language itself:

Language corresponds to properties of the natural

unambiguous as it corresponds to properties of the natural

for certain formal languages, ambiguity is

the string is ambiguous:

When more than one tree exists for some string, we say that

which they apply

This can happen only when the derivations differ in the rules

string (correspond to different trees.

Sometimes, however, different derivations (of the same

Correspondence between trees and derivations

This tree represents the following derivations (among others):

(1) Np Np Np

(2) the car in the N

(3) the car in the hat

(4) Np D

A

b
A context-free grammar for natural languages:

```
S -> VP
S -> NP V
VP -> NP VP
VP -> V NP
NP -> N NP
NP -> N
V -> saw
V -> loves
V -> smile
N -> in
N -> sleeps
N -> the
```

Ambiguity: This situation is known as syntactic or structural ambiguity. This occurs when there is more than one possible derivation for a sentence.

Example: "The cat in the hat." The phrase "in the hat" can be derived from the sentence in two different ways, depending on the order of the phrase.

Using linguistic terminology, in the left tree the second occurrence of the prepositional phrase in the hat modifies the noun phrase the cat, whereas in the right tree, the hat modifies the noun phrase the cat in the hat.
rules accordingly:

\[ V \in S, V'p, VpS, VpP, \text{ etc.} \]

We must also change the set of categories so that we get the \( Dp \)'s, \( DpD \), \( Np \)'s, \( NpD \), \( NpP \), \( NpPp \) etc. We must also change the set of non-terminal symbols that are assigned for them. Features in the non-terminal that is assigned for them are features that are not terminal symbols such that categories that must agree reflect agreement.

Agreement

Both problems are easy to solve.

1. Over-generation.

Over-generation of verb forms of verbs such as "smile" as "smiles". This is another case of plural form of verb such as "she".

2. There is no agreement of subject-verb agreement.

Verbs generate strings that are not in the intended language. It is a case we say that the grammar overgenerates. Giving an abstract phrase structure for English is a transform over words such as love might occur without case. In a transform over words such as love might occur with a noun phrase complement while some subcategorization of verbs, and an intrasubjective verb such as "steep" might occur with a noun phrase without case. It ignores the valence of verbs; there is no distinction in agreement.

There are two major problems with this grammar:

Context-free grammars for natural languages

Verbs

The cat steps.

\[ S \rightarrow Np \rightarrow Vp \rightarrow VpPp \]

A derivation tree for the cat steps is:

The augmented grammar can derive strings such as the cat
every natural language?

Are context-free grammars sufficient for generating
languages? Are there natural languages that are not context-

The interesting question, of course, involves natural

languages generated by context-free grammars. Therefore, there are certain sets of strings that cannot be

However, some (formal) languages are not context-free, and

unbounded dependencies, extraction, extraposition, etc.

Constructions including some non-trivial phenomena such as

context-free grammars can be used for a variety of syntactic

Context-free grammars for natural languages

Agreement

\[ \begin{align*}
& S \rightarrow N p l \text{ Np} \text{ Dp} \\
& V p s \rightarrow V p s \text{ Vp} \text{ Np} \\
& D p \rightarrow D p \\
& N p s \rightarrow N p s \text{ Np} \\
& D p l \rightarrow D p l \\
& S \rightarrow N p s \text{ Vp} \\
& N p l \rightarrow N p l \\
& V p s \rightarrow V p s \\
& \text{many} \rightarrow D p l \\
& \text{a} \rightarrow D s e \text{ a} \\
& \text{see} \rightarrow V p l \\
& \text{love} \rightarrow V p l \\
& \text{smile} \rightarrow V p l \\
& \text{sleep} \rightarrow V p l \\
& \text{in} \rightarrow p \\
& \text{has} \rightarrow N p l \\
& \text{cat} \rightarrow N p l \\
& \text{is} \rightarrow N s e \\
& \text{are} \rightarrow N s e \\
& \text{are} \rightarrow N s e \\
\end{align*} \]