Chapter II : Context-Free Languages

Section 1 : Context-free grammars
Section 2: Chomski normal form
Section 3: Pusdown automata
Section 4: Equivalence of CFG and PDA
Section 5: Non-context-free languages
Section 6: Pumping lemma for context-free languages

Context-Free Grammars

Regular languages
- Finite automata (DFA, NFA, GNFA)
- Regular expressions

Observe that the regular expressions that describe infinite languages encode certain types of recursive definitions by the star operator

$$0^*101^* \ni 10,\ 010,\ 0010,\ 00010,\ \ldots,\ 00\ldots010$$
$$\ni 00\ldots0101,\ 00\ldots01011,\ \ldots,\ 00\ldots01011\ldots1$$

A context-free grammar is a more powerfull method of describing languages with certain recursive syntactic features.

Context-free grammars $\Longrightarrow$ Context-free languages $\supsetneq$ Regular Languages

G_1: $A \rightarrow 0A1$
$A \rightarrow B$
$B \rightarrow \#$

A grammar consists of

- a collection of substitution rules (productions, rewriting rules)
- each rule is a line in the grammar formed by:
  - a symbol – variable
  - a string of symbols consisting of variables and terminals
- one variable is designated as the start variable.

In G_1: variables – $\{A, B\}$
terminals – $\{0, 1, \#\}$
start variable – A

G_1: $A \rightarrow 0A1$
$A \rightarrow B$
$B \rightarrow \#$

variables $- \{A, B\}$
terminals $- \{0, 1, \#\}$
start variable $-$ A

A grammar generates all the strings in a language $\Rightarrow \mathcal{L}(G)$

- we start with the start variable and replace in its product some of the variables with some of their products
- we repeat this procedure until we have no more variables, i.e., we get a string of terminal symbols.

The sequence of substitutions to obtain a string is called a <u>derivation</u>.

$A \Rightarrow B \Rightarrow \#$

$A \Rightarrow 0A1 \Rightarrow 0B1 \Rightarrow 0\#1$

$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 00B11 \Rightarrow 00\#11$

$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$

G_1:

$$A \to 0A1$$
$$A \to B$$
$$B \to \#$$

variables — $\{A, B\}$

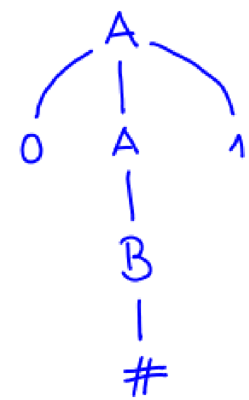
terminals — $\{0, 1, \#\}$

start variable — A

Alternatively, a derivation can be represented as a *parse tree*.

$A \Rightarrow B \Rightarrow \#$

$A \Rightarrow 0A1 \Rightarrow 0B1 \Rightarrow 0\#1$



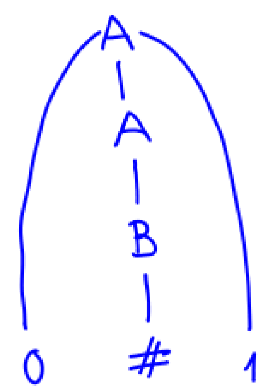
G_1: $A \to 0A1$
$A \to B$
$B \to \#$

variables – $\{A, B\}$
terminals – $\{0, 1, \#\}$
start variable – A

Alternatively, a derivation can be represented as a <u>parse tree</u>.

$A \Rightarrow B \Rightarrow \#$

$A \Rightarrow 0A1 \Rightarrow 0B1 \Rightarrow 0\#1$



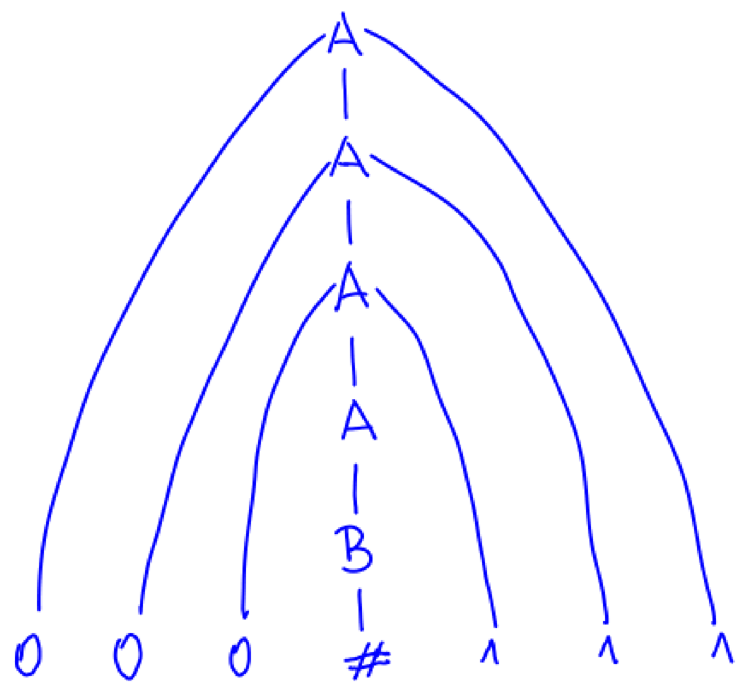
G_1:

$A \to 0A1$
$A \to B$
$B \to \#$

variables – $\{A, B\}$
terminals – $\{0, 1, \#\}$
start variable – A

Alternatively, a derivation can be represented as a <u>parse tree</u>.

$$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$$



G_1: $A \to 0A1$
$A \to B$
$B \to \#$

variables – $\{A, B\}$
terminals – $\{0, 1, \#\}$
start variable – A

$A \Rightarrow B \Rightarrow \#$

$A \Rightarrow 0A1 \Rightarrow 0B1 \Rightarrow 0\#1$

$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 00B11 \Rightarrow 00\#11$

$A \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 000A111 \Rightarrow 000B111 \Rightarrow 000\#111$

All the strings generated in this way from a given context-free grammar G constitute the <u>language of the grammar G</u>, $\mathcal{L}(G)$.

Observe that $\mathcal{L}(G_1) = \{0^n \# 1^n \,/\, n \geq 0\}$.

A language generated by a context-free language is called a context-free language (CFL)

Reduced notation for grammars

G_1: $A \to 0A1$
$A \to B$
$B \to \#$

$\Longrightarrow$

G_1: $A \to 0A1 \mid B$
$B \to \#$

Let G_2 – a fragment of English language

⟨Sentence⟩ → ⟨Noun-phrase⟩⟨verb-phrase⟩

⟨Noun-phrase⟩ → ⟨Cmplx-noun⟩ | ⟨Cmplx-noun⟩⟨prep-phrase⟩

⟨verb-phrase⟩ → ⟨Cmplx-verb⟩ | ⟨Cmplx-verb⟩⟨prep-phrase⟩

⟨prep-phrase⟩ → ⟨prep⟩⟨Cmplx-noun⟩

⟨Cmplx-noun⟩ → ⟨article⟩⟨Noun⟩

⟨Cmplx-verb⟩ → ⟨verb⟩ | ⟨verb⟩⟨Noun-phrase⟩

⟨Article⟩ → a | the

⟨Noun⟩ → boy | girl | flower

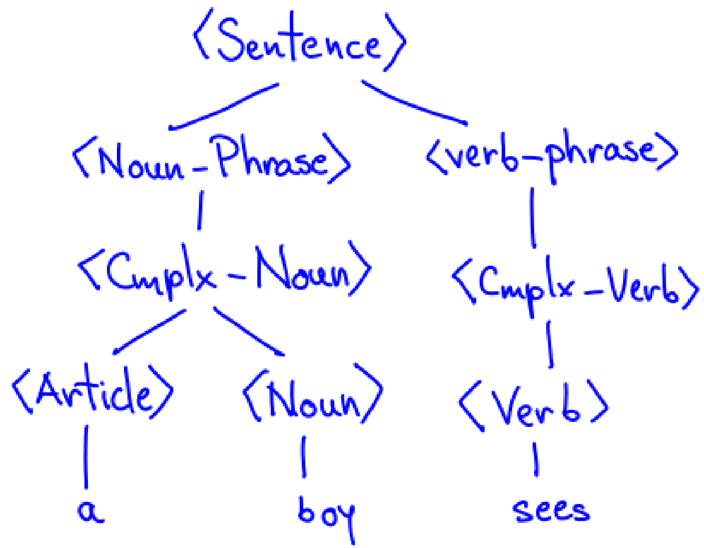
⟨verb⟩ → touches | likes | sees

⟨Prep⟩ → with

Give the derivation of the phrase "a boy sees"

<Sentence> => <Noun-Phrase><Verb-Phrase>
=> <Cmplx-Noun><Verb-Phrase>
=> <Article><Noun><Verb-Phrase>
=> a <Noun><Verb-Phrase>
=> a boy <Verb-Phrase>
=> a boy <Cmplx-Verb>
=> a boy <Verb>
=> a boy sees

Draw the parse tree for the previous phrase



<u>Exercise</u>: Find derivations and draw the corresponding parse trees for the sentences

"the boy sees a flower"

"a girl with a flower likes the boy"

Definition [Context-free grammar]:

A context-free grammar is a tuple $G=(V,\Sigma,R,S)$, where

- V is a finite set of variables
- Σ is a finite set of terminals , $V \cap \Sigma = \emptyset$
- R is a finite set of rules
  - a rule is composed from a variable and a string from $V \cup \Sigma$
- $S \in V$ is the start variable.

If $u,v,w \in (V \cup \Sigma)^*$ and $A \to w \in R$, we say that uAv yields uwv.
and write $uAv \Rightarrow uwv$

We say that u derives v, written $u \Rightarrow^* v$ if

- either $u = v$ or
- there exist $u_1, \dots u_k \in (V \cup \Sigma)^*$ s.t.
  $u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow \dots \Rightarrow u_k \Rightarrow v$

The language generated by G is $\mathcal{L}(G) = \{ w \in \Sigma^* / S \Rightarrow^* w \}$

The grammar for the fragment of English can be described as

$$G = (V, \Sigma, R, S), \text{ where}$$

- $V = \{$ ⟨Sentence⟩, ⟨noun-phrase⟩, ⟨verb-phrase⟩, ⟨prep-phrase⟩, ⟨Cmplx-noun⟩, ⟨Cmplx-verb⟩, ⟨Article⟩, ⟨Noun⟩, ⟨Verb⟩, ⟨Prep⟩ $\}$
- $\Sigma = \{$ a, the, boy, girl, flower, touches, likes, sees, with $\}$

or alternatively,

$$\Sigma = \{a, b, c, \ldots, x, y, z\}$$

- $S =$ ⟨Sentence⟩

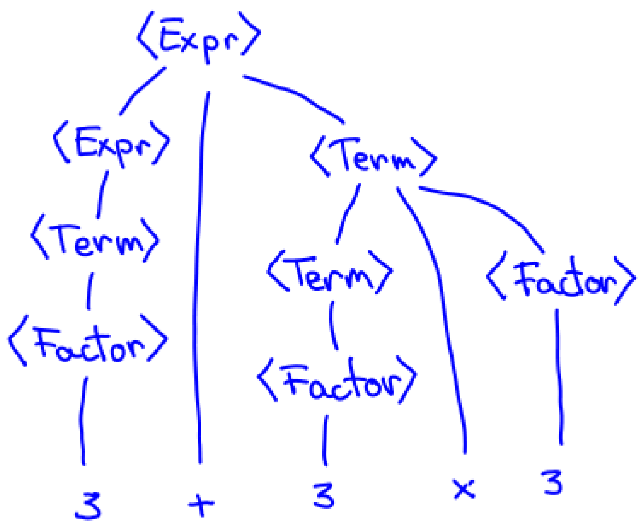
Let $G_3 = (V, \Sigma, R, \langle \text{Expr} \rangle)$ where

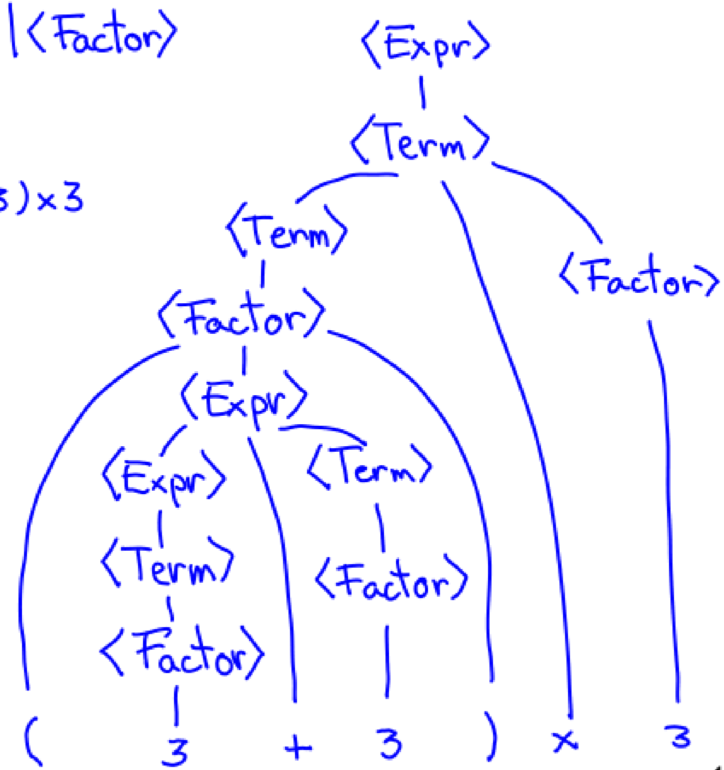
$V = \{\langle \text{Expr} \rangle, \langle \text{Term} \rangle, \langle \text{Factor} \rangle\}$

$\Sigma = \{3, +, \times, (,)\}$

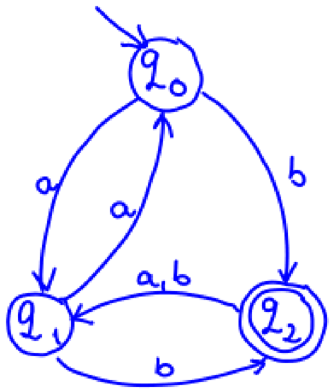
$$
\begin{aligned}
R:\quad \langle \text{Expr} \rangle &\rightarrow \langle \text{Expr} \rangle + \langle \text{Term} \rangle \mid \langle \text{Term} \rangle \\
\langle \text{Term} \rangle &\rightarrow \langle \text{Term} \rangle \times \langle \text{Factor} \rangle \mid \langle \text{Factor} \rangle \\
\langle \text{Factor} \rangle &\rightarrow (\langle \text{Expr} \rangle) \mid 3
\end{aligned}
$$

Parse trees for $3+3\times 3$ and $(3+3)\times 3$





Theorem: Any DFA can be converted into an equivalent CFG.



δ	a	b
q_0	q_1	q_2
q_1	q_0	q_2
q_2	q_1	q_1

Consider the variables:

R_0 – for q_0

R_1 – for q_1

R_2 – for q_2

Rules:

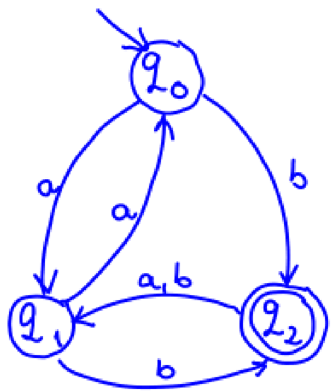
initiality: $R_0 \longrightarrow aR_1 \mid bR_2$

$R_1 \longrightarrow aR_0 \mid bR_2$

$R_2 \longrightarrow aR_1 \mid bR_1$

finality: $R_2 \longrightarrow \varepsilon$

Theorem: Any DFA can be converted into an equivalent CFG.



δ	a	b
q_0	q_1	q_2
q_1	q_0	q_2
q_2	q_1	q_1

$G = (V, \Sigma, \mathcal{R}, R_0)$

$V = \{R_0, R_1, R_2\}$

$\Sigma = \{a, b\}$

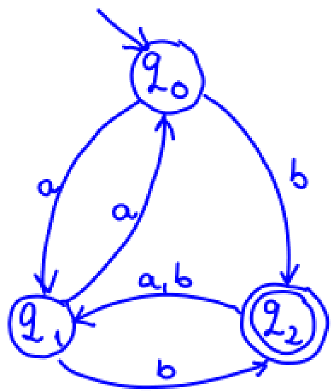
$\mathcal{R}$:

$R_0 \rightarrow aR_1 \mid bR_2$

$R_1 \rightarrow aR_0 \mid bR_2$

$R_2 \rightarrow aR_1 \mid bR_1 \mid \varepsilon$

Theorem: Any DFA can be converted into an equivalent CFG.



input aabab

$$\rightarrow q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_0 \xrightarrow{b} q_2 \xrightarrow{a} q_1 \xrightarrow{b} q_2$$

derivation of aabab

$$\begin{aligned} R_0 &\Rightarrow aR_1 \\ &\Rightarrow aaR_0 \\ &\Rightarrow aabR_2 \\ &\Rightarrow aabaR_1 \\ &\Rightarrow aababR_2 \\ &\Rightarrow aabab\varepsilon \\ &= aabab \end{aligned}$$

$\mathcal{G}$:

$$\begin{aligned} R_0 &\rightarrow aR_1 \mid bR_2 \\ R_1 &\rightarrow aR_0 \mid bR_2 \\ R_2 &\rightarrow aR_1 \mid bR_1 \mid \varepsilon \end{aligned}$$

Theorem: Any DFA can be converted into an equivalent CFG.

Consider the DFA $D=(Q, \Sigma, \delta, q_0, F)$

We construct the equivalent CFG

$$G=(V, \Sigma, \mathcal{R}, R_0)\text{, where}$$

- If $Q=\{q_0, q_1, \dots q_k\}$, $V=\{R_0, R_1, \dots R_k\}$
- $\mathcal{R}$ is defined, for each $i=\overline{0,k}$ as follows

  if $\delta(q_i, a)=q_j$, $\mathcal{R}$ contains the rule

  $$R_i \longrightarrow aR_j$$

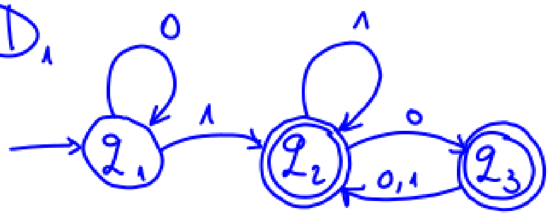
  for each $q_f \in F$, $\mathcal{R}$ contains the rule

  $$R_f \longrightarrow \varepsilon$$

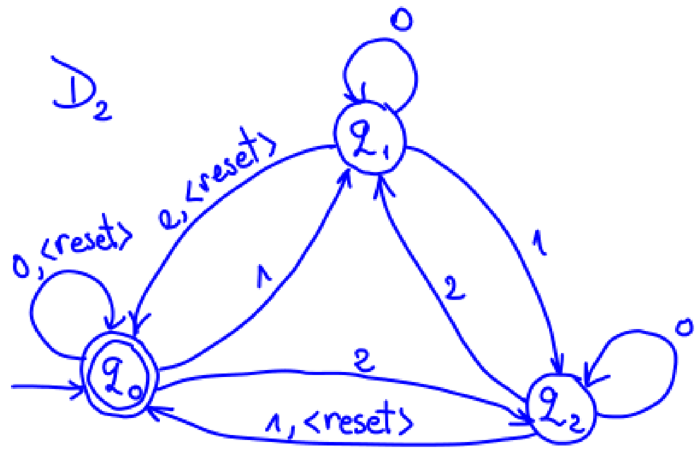
- R_0 that corresponds to the initial state q_0 is the start variable.

Construct an equivalent CFG for the following DFAs.

D_1



D_2



Corollary: Any regular language is a context-free language.

Proof: Let L be a regular language.

Then, there exists a DFA D that recognizes L.

From the previous Theorem we know that we can construct a CFG G such that $\mathcal{L}(D) = \mathcal{L}(G)$

Since $\mathcal{L}(D) = L$ $\Rightarrow$

$\Rightarrow$ there exists a CFG G that generates L.

Hence, L is a context-free language.

Corollary: Any regular language is a context-free language.

Theorem: There exist context-free languages that are not regular.

Proof: The language $L = \{0^n 1^n / n \geq 0\}$ is not a regular language.

We construct a CFG $G = (V, \Sigma, R, S)$ where

$V = \{S\}$
$\Sigma = \{0, 1\}$
$R: \quad S \longrightarrow 0S1 \mid \varepsilon$

Observe that:

$\mathcal{L}(G) \ni \varepsilon : \quad S \Rightarrow \varepsilon$

$\mathcal{L}(G) \ni 01 : \quad S \Rightarrow 0S1 \Rightarrow 0\varepsilon 1 = 01$

$\mathcal{L}(G) \ni 0^2 1^2 : \quad S \Rightarrow 0S1 \Rightarrow 00S11 \Rightarrow 00\varepsilon 11 \Rightarrow 0^2 1^2$

...

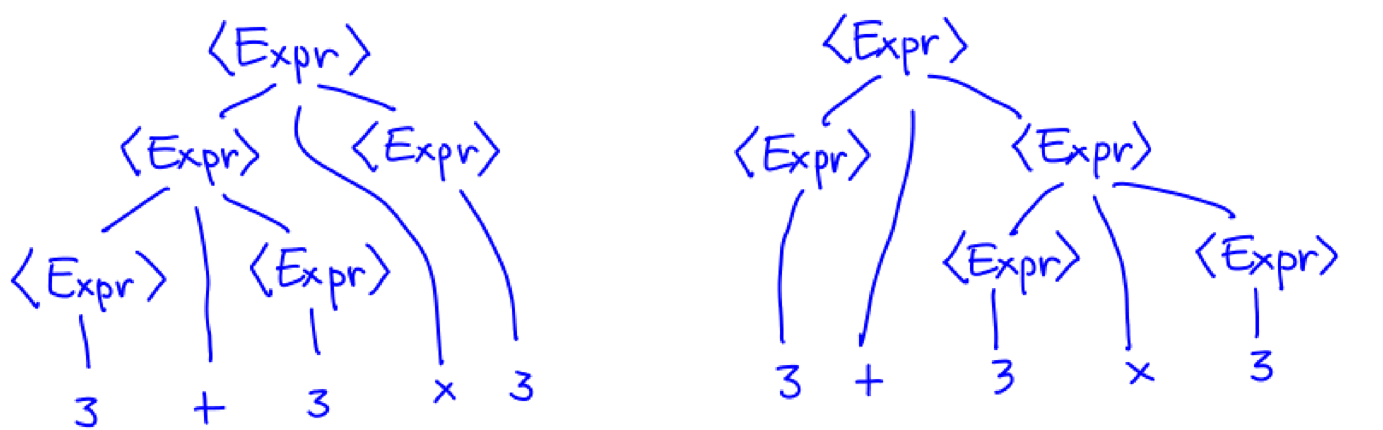
$\mathcal{L}(G) \ni 0^n 1^n : \quad S \Rightarrow 0S1 \Rightarrow 00S11 \Rightarrow \dots \underbrace{0..0}_{n} S \underbrace{1..1}_{n} \Rightarrow 0^n \varepsilon 1^n = 0^n 1^n$

Ambiguity

Consider the grammar $G_4 = (\{\langle Expr\rangle\}, \{+, \times, 3, (,)\}, R, \langle Expr\rangle)$

R: $\langle Expr\rangle \rightarrow \langle Expr\rangle + \langle Expr\rangle \mid \langle Expr\rangle \times \langle Expr\rangle \mid (\langle Expr\rangle) \mid 3$

Two parse trees for the string $3 + 3 \times 3$



G_4 is <u>ambiguous</u> – it generates $3 + 3 \times 3$ in two different ways.
If later we will associate some semantics to the strings in $\mathcal{L}(G)$, $3 + 3 \times 3$ will have two different meanings.

Ambiguity

Consider the grammar G_2 of the fragment of English.

Prove that the following phrase is ambiguous

the girl touches the boy with the flower

Hint:

the girl touches <u>the boy with the flower</u>

the girl <u>touches</u> the boy <u>with the flower</u>

Ambiguity

If a grammar can generate the same string in different ways, that string will have different parse trees and thus different meanings.

This is undesirable for programming languages, where a given program should have a unique interpretation.

Definition: A string w is derived ambiguously from the CFG G if it has two or more different left-most* derivations.
A grammar G is ambiguous if it generates some string ambiguously.

---

*A left-most derivation is a derivation in which we always replace the left-most variable.

Ambiguity

Some context-free languages can only be generated by ambiguous grammars. They are called <u>inherently ambiguous languages</u>

Consider the language $L = \{a^i b^j c^k \mid i=j \text{ or } j=k,\ i,j,k \geq 0\}$

- Provide a CFG that generates L.

$G = (V, \Sigma, R, S)$

$V = \{S, A, A', C, C'\}$

$\Sigma = \{a, b, c\}$

R:

$$
\begin{aligned}
S &\to A \mid C \\
A &\to A' \mid Ac \\
C &\to C' \mid aC \\
A' &\to aA'b \mid \varepsilon \\
C' &\to bC'c \mid \varepsilon
\end{aligned}
$$

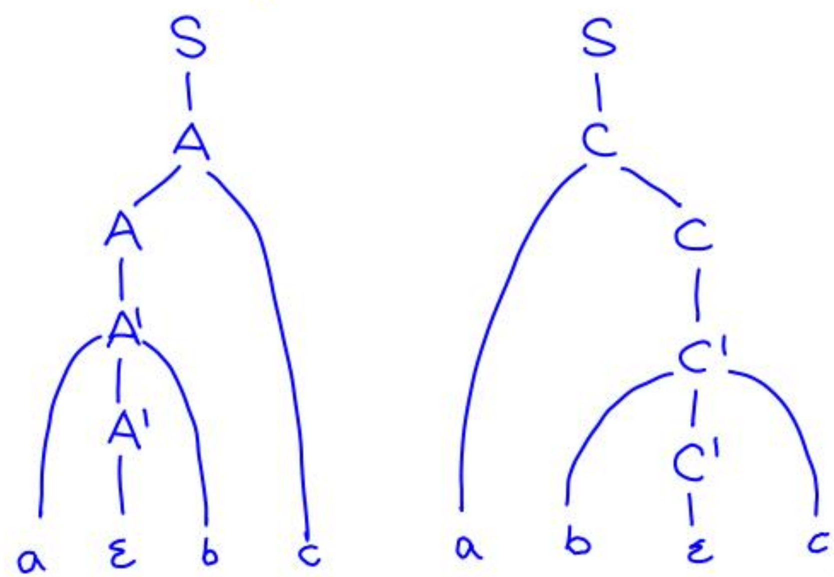
Ambiguity

Some context-free languages can only be generated by ambiguous grammars. They are called <u>inherently ambiguous languages</u>

Consider the language $L = \{a^i b^j c^k \,/\, i=j \text{ or } j=k \,,\ i,j,k \geq 0\}$

- Provide a CFG that generates L.
- Is the generated CFG ambiguous?

R:
$$
\begin{aligned}
S &\to A \mid C \\
A &\to A' \mid Ac \\
C &\to C' \mid aC \\
A' &\to aA'b \mid \varepsilon \\
C' &\to bC'c \mid \varepsilon
\end{aligned}
$$



Ambiguity

Some context-free languages can only be generated by ambiguous grammars. They are called inherently ambiguous languages

Consider the language $L = \{a^i b^j c^k \mid i=j \text{ or } j=k,\ i,j,k \geq 0\}$

- Provide a CFG that generates L.
- Is the generated CFG ambiguous?
- Is L inherently ambiguous? Motivate your answer.