Context-free Grammar (CFG)

A CFG consists of the following components:

- A set of variables usually denoted by capital letters e.g. $\{A, B, C\}$
- An alphabet Σ: The alphabet letters are called terminals e.g. $\Sigma = \{0, 1\}$
- A set of rules of the form e.g.

start

$A \to BCB$
$B \to 0B$
$B \to \varepsilon$
$C \to 1C$
$C \to \varepsilon$

variables — strings of variables and terminals or ε

- A start variable
  (usually the LHS of the first row)

Use CFG to generate a language

① Start with the start variable

② A rule of the form var $\to w$ tells us that we can take one of the occurances of the var and replace it with w

③ We're done when the string has no variable

e.g. $A \Rightarrow BCB \Rightarrow 0BCB \Rightarrow 00BCB \Rightarrow 00BC \Rightarrow 00B1C \Rightarrow 00\,1C \Rightarrow 001$

Notation A CFG is a 4-tuple (V, Σ, R, A) — $A \in V$ start.

R: $S \to w$ where $S \in V$, $w \in (V \cup \Sigma \cup \{\varepsilon\})^*$

- $w \Rightarrow u$ if u can be obtained from w by applying one rule.
- $w \overset{*}{\Rightarrow} u$ if u can be obtained from w by applying a sequence of rules

DEF The language of a CFG with a start variable A is

$$\{w \in \Sigma^* \mid A \overset{*}{\Rightarrow} w\}$$

Example

① $A \to 0A1$
$A \to \varepsilon$

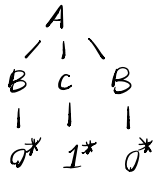
$\Rightarrow \{0^n 1^n \mid n \geq 0\}$

② $A \to BCB$
$B \to 0B$
$B \to \varepsilon$
$C \to 1C$
$C \to \varepsilon$

$\{ w \in \Sigma^* \mid B \Rightarrow^* w \} = 0^*$

$\{ w \in \Sigma^* \mid C \Rightarrow^* w \} = 1^*$

$\Rightarrow \quad 0^* 1^* 0^*$



③ Design a CFG whose language is

$\{ w \mid w$ starts and ends with the same symbol, $|w| \geq 2 \}$ $\quad \Sigma = \{0,1\}$

$A \to 0B0$
$A \to 1B1$

$B \to \varepsilon$
$B \to 0B$
$B \to 1B$

④ Palindromes $= \{ w \mid w = w^R \}$ over $\Sigma = \{0,1\}$

$A \to 0A0$
$A \to 1A1$

$A \to 0$
$A \to 1$
$A \to \varepsilon$

⑤ $\{ w \mid$ number of 1 is divisible by 2 $\}$

$A \to A1A1A$
$A \to \varepsilon$
$A \to 0A$

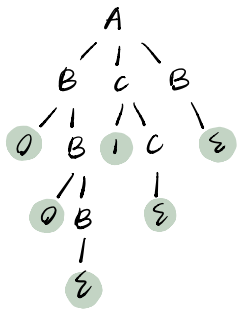
The first rule can applied repeatedly to generate an

$\underbrace{A1A1A1 \cdots 1A1}_{\text{even number of A's}}$

Then we can use $A \Rightarrow^* 0^*$ to replace those A with 0's.

Parsing tree

$A \Rightarrow BCB \Rightarrow 0BCB \Rightarrow 00BCB \Rightarrow 00BC \Rightarrow 00B1C \Rightarrow 00 1C \Rightarrow 001$



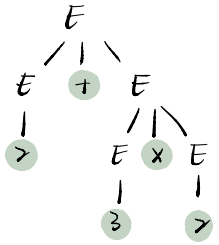
Notation $\quad \left.\begin{array}{l} A \to BC \\ A \to \varepsilon \\ A \to AB1 \end{array}\right\} \Leftrightarrow A \to BC \mid \varepsilon \mid AB1$

Consider the following CFG $\Sigma = \{+, \times, 0, \dots, 9\}$

$E \to E+E \mid E \times E \mid 0 \mid 1 \mid 2 \mid \dots \mid 9$

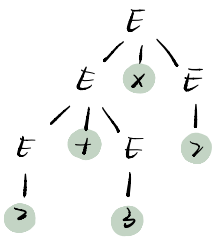
and the derivation

$E \Rightarrow E+E \Rightarrow 2+E \Rightarrow 2+E \times E \Rightarrow^* 2+3\times 2$



We can also have

$E \Rightarrow E \times E \Rightarrow E+E \times E \Rightarrow^* 2+3\times 2$



same word, different parsing trees

The CFG $E \to E+E \mid E \times E \mid 0 \mid 1 \mid 2 \mid \dots \mid 9$ is ambiguous as some words such as $2+3\times 2$ have 2 different parsing trees.

Remark

$\begin{cases} E \Rightarrow E+E \Rightarrow E+E \times E \Rightarrow 2+E \times E \Rightarrow 2+3 \times E \Rightarrow 2+3\times 2 \\ E \Rightarrow E+E \Rightarrow 2+E \Rightarrow 2+E \times E \Rightarrow 2+3 \times E \Rightarrow 2+3\times 2 \end{cases}$

have identical parsing trees.

DEF A left-most derivation for a word is a derivation where we always apply the rule to the left-most variable of the string.

DEF A CFG is ambiguous if there is a word with 2 different left-most derivations.

Sometimes, we can fix the ambiguouty:

$E \to E+T \mid T$
$T \to T \times F \mid F$
$F \to 0 \mid 1 \mid \cdots \mid 9$

↑ non ambiguous

$\equiv$

$E \to E+E$
$E \to E \times E$
$E \to 0 \mid 1 \mid \cdots \mid 9$

↑ ambiguous

Remark There are CFL that are inherily ambiguous;
i.e. They cannot be recognized by any non-ambiguous CFG's.

e.g. $\{a^i b^j c^k \mid i=j \text{ or } j=k\}$ see book.

$A \to BC \mid DE$
$C \to cC \mid \varepsilon$
$B \to aBb \mid \varepsilon$

$D \to aD \mid \varepsilon$
$E \to bEc \mid \varepsilon$

THM Every regular language is recognized by a CFG.

Since the language is regular, it is described by a regular expression R.

We will use induction on the length of R.

base case
① $R = a$ for some $a \in \Sigma$ $\quad A \to a$
② $R = \varepsilon$ $\quad A \to \varepsilon$
③ $R = \phi$

inductive step Assume that every R of length $< m$ can be converted to a CFG.

① $R = (R_1)^*$ where $|R_1| < m$
Add $A \to AA, A \to \varepsilon$ to the CFG of R_1.

② $R = R_1 \cup R_2$
Add $S \to A_1 \mid A_2$ to the CFG of R_1, R_2
(S: new start; $A_1 \to$; $A_2 \to$)

③ $R = R_1 R_2$
Add $S \to A_1 A_2$

Chomsky Normal Form

Every CFG can be converted to an equivalent CFG with the start state A s.t. every rule is of the following form:

$B \to CD$ where $B, C, D \in V$, $C, D \neq A$

$B \to a$ where $B \in V$, $a \in \Sigma$

$A \to \varepsilon$ only start can go to ε

(A) ← start.

Ex. Which one in Chomsky normal form?

$A \to \varepsilon$	$A \to \varepsilon$	$A \to BC$
$A \to AB$ ✗	$A \to BC$	$B \to BC$
$B \to 0$	$B \to 0B$ ✗	$B \to DB$
$A \to 1$	$C \to 1$	$C \to 1$
		$D \to 0$
		$L = \{\}$

Convert an arbitrary CFG (V, Σ, R, A) into Chomsky Normal Form:

① Add a new start variable A_0 and the rule $A_0 \to A$

⇒ Guarantee that A_0 is never in the RHS

② For every $b \in \Sigma$, add a new variable B and replace all the occurance of b in the RHS with B. Finally, add $B \to b$

③ If there is a rule $B \to \varepsilon$ where $B \neq A_0$, then we remove it, and for every rule $C \to u_1 B u_2 B \cdots B u_{k+1}$ (k B's), we add all the possible 2^k combinations that can be obtained from the RHS by removing some of the B's.

④ Consider $B \to C_1 C_2 \cdots C_k$ where $C_1 \cdots C_k \in V$ and $k \geq 2$:

$B \to C_1 D_1$

$D_1 \to C_2 D_2$

$D_2 \to C_3 D_3$

$\vdots$

$D_{k-2} \to C_{k-1} C_k$

where $D_1, D_2 \cdots D_{k-2}$ are new variables

⑤ $B \to C$ we remove this and for every $C \to u$, we add $B \to u$. We repeat this until we eliminate all such rules.

Push-Down Automaton

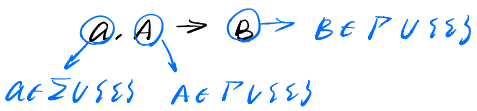
Recall: Stack (last in first out)

PDA ≡ NFA + infinite stack

DEF A PDA is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ (Γ: stack alphabet)

Everything is similar to NFA except:

- The label on the arrows are of the form

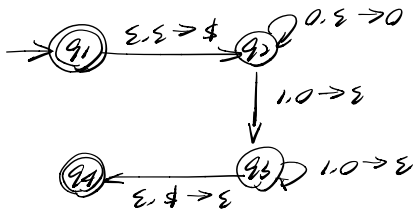


which says if we need a from input, the pop A and push B to stack

$$\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \to P(Q \times \Gamma_\varepsilon) \quad \text{where } \Sigma_\varepsilon = \Sigma \cup \{\varepsilon\},\ \Gamma_\varepsilon = \Gamma \cup \{\varepsilon\}$$

- Acceptance: A PDA accepts a word w if it can read all of w and end at an accept state. (No need to empty stack) (non-deterministic)

Ex. $\Sigma = \{0, 1\}$ $\Gamma = \{0, \$\}$



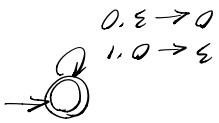
Accept 0011:

$q_1, \sqcup \to q_2, \$ \to q_2, 0\$ \to q_2, 00\$ \to q_3, 0\$ \to q_3, \$ \to q_4, \sqcup$ accept

$L = \{0^n 1^n \mid n \geq 0\}$

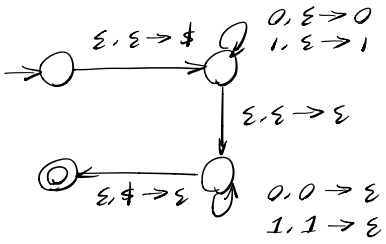
Ex. Design a PDA with one state for the following language L

$\{w \mid$ no prefix of w contains more 1's than 0's $\}$



Ex. Design a PDA for $L = \{ww^R \mid w \in \Sigma^*\}$, $\Sigma = \{0,1\}$

We use $ to mark the beginning of the stack



non-deterministic

CFG ≡ PDA

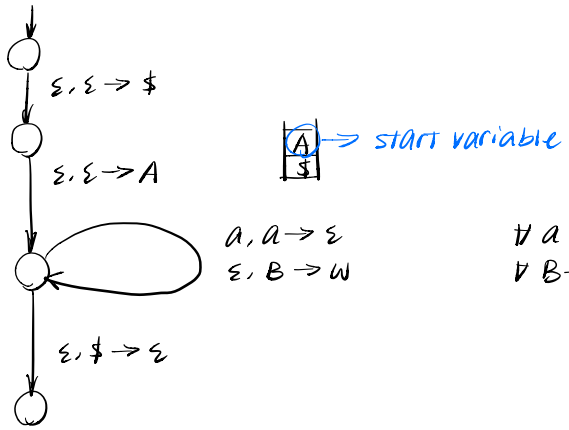
THM If L is recognized by a CFG, then there is a PDA for L.

Since L is context free, there is a CFG in Chomskey normal form that generates L

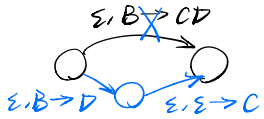
(V, Σ, R, A)

We start by constructing something that is not quite a PDA: It can pop more than one character at a time

Let $ be a symbol $\$ \notin V \cup \Sigma \cup \{\varepsilon\}$

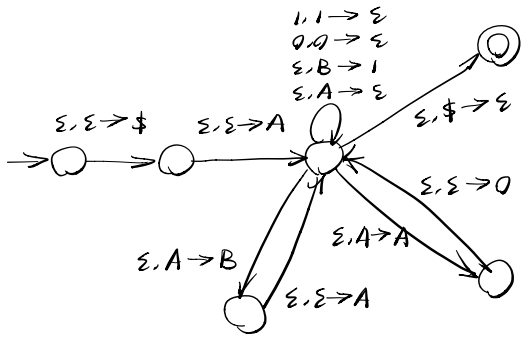


How to fix $B \to CD$?



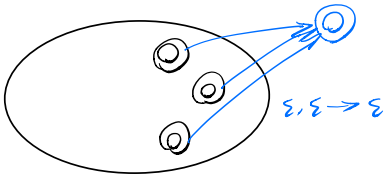
e.g. $A \to AB \mid 0A \mid \varepsilon$

$B \to 1$



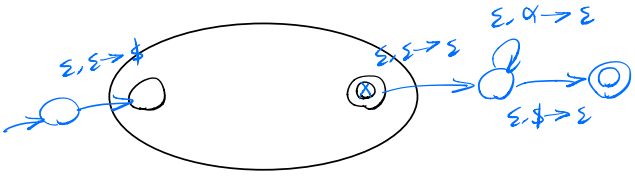
THM *If L is recognized by a PDA, then there is a CFG for L.*

① a single accept state

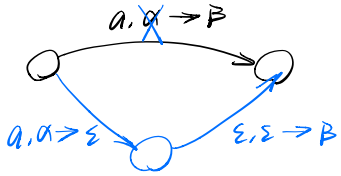


② It empties the stack before accepting

Pick a new symbol $



③ In each transition $a, \alpha \to \beta$ either $\alpha = \varepsilon$ or $\beta = \varepsilon$
(don't push and pop at same time)



Variables For every two states p, q in PDA, we have a variable $A_{p,q}$

It is supposed to generate all the words w that can take us from p to q starting and ending with the empty stack.

$$A_{p,q} \Rightarrow^* w$$

Start $A_{q_{start}, q_{accept}}$

$\Rightarrow^*$ the words from start to accept with an empty stack
i.e. Language of the PDA

Rule

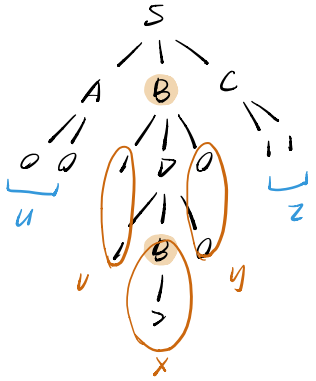
- $A_{p,p} \to \varepsilon$
- $A_{p,q} \to A_{p,r} A_{r,q}$ (uw: u, w)
- 

$A_{pq} \to a A_{rs} b$

Languages that are NOT context-free

Ex. $S \to ABC$
$A \to 00$
$C \to 11$
$B \to 1D0 \mid 2$
$D \to 1B0$

$w = 0011200111$



$B \Rightarrow^* 11B00$

$B \Rightarrow^* 1111B0000$

$B \Rightarrow^* 111111B000000$

Thus, $0011111120000001111 \in L$

How long w has to be to guarantee that we will find such R?

Let $G = (V, \Sigma, R, A)$ for L. Let b be the longest length of the RHS of a rule: $B \to \underbrace{\cdots}_{\text{length } b}$

Observation 1 Each rule has at most b children

Observation 2 If no such R then the length of the longest path from A to a leaf is at most $|V|+1$

$\Rightarrow$ at most $b^{|V|}$ leaves

Pumping Lemma of CFG

If L is a CFG, then there $\exists$ a positive p (called pumping constant) such that if $w \in L$ and $|w| \geq p$ then there $\exists$ a decomposition

$$w = uvxyz \quad \text{s.t.}$$

① $|vxy| \leq p$

② at least one of v and y is not ε

③ $uv^i xy^i z \in L$

Ex. Show that $L = \{a^n b^n c^n \mid n \geq 0\}$ over $\Sigma = \{a, b, c\}$ is NOT context free.

Suppose it is, Let p be its pumping constant.
Consider $w = a^p b^p c^p \in L$
By pumping lemma: $w = uvxyz$ s.t. $\cdots$

Since $|vxy| \leq p$, at least one of the letters a, b, c is missing in vxy.
Then in $uvvxyyz$, at least one of the other two symbols appears more than the missing symbols in xyz. ↯

Ex. Show that both

$$L_1 = \{a^n b^n c^m \mid n, m \geq 0\}$$

$$L_2 = \{a^m b^n c^n \mid n, m \geq 0\}$$

are context-free.

for L_1, $A \to BC$, $B \to aBb \mid \varepsilon$, $C \to cC \mid \varepsilon$

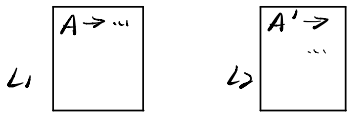
for L_2, $A \to BC$, $B \to bB \mid \varepsilon$, $C \to bCc \mid \varepsilon$

Note, $\{a^n b^n c^n \mid n \geq 0\} = \{a^n b^n c^m \mid n, m \geq 0\} \cap \{a^m b^n c^n \mid n, m \geq 0\}$

NOT CFG — CFG — CFG

⇒ CFG NOT closed under intersection

However, CFG is closed under union and concatenation.



But NOT closed under complementation

Take a non-context-free language s.t. $L = L_1 \cap L_2$ where L_1 and L_2 are context-free. ⇒ $L^c = (L_1^c \cup L_2^c)$ will be context free ↯

Ex. Show that $\{ww \mid w \in \{0,1\}^*\}$ is NOT context-free over $\Sigma = \{0,1\}$

Note: $\{ww^R \mid w \in \{0,1\}^*\}$ ✓

$w = 0^p 1^p 0^p 1^p \in L$

$\underbrace{000\cdots 111\cdots 1}_{\text{first half}} 00\cdots 011\cdots 1$

① vxy completely in the first half

Then $uvvxyyz$ is of length $4p + |vy|$.

start with 0 — start with y

$00\cdots 11\cdots 1110\cdots 01\cdots 1$

$|vxy| > p$

② vxy completely in the second half

end with 0 — end with 1

$00\cdots 011\cdots 100\cdots 011\cdots 1$

② vxy touches both sides of the first and second half

$$00\cdots0\,11\cdots1\,00\cdots0\,11\cdots1$$

then $i=0 \Rightarrow uxz \in L$

if $uxz = 0^k1^k0^k1^k \Rightarrow k=p$ because the word starts with $\underbrace{0\cdots01}_{p}$

but $uxz = 0^p1^p0^p1^p = uvxyz = w$ ↯

Ex. $\{ww \mid w \text{ has same number of } a,b,c\}$ over $\Sigma=\{a,b,c\}$ is NOT context free.

$w = a^p b^p c^p$