Context-Free Grammars and PDAs

Theorem 4.6. *Let Σ be an alphabet, and let L be a language over L. Then L is context-free if and only if there is a pushdown automaton whose input alphabet is Σ such that $L = L(M)$.*

- construction of a pushdown automaton for *L*

Suppose that L is a context-free language over an alphabet Σ. Let $G = (V, \Sigma, P, S)$ be a context-free grammar for L. Then we can construct a pushdown automaton M that accepts L. In fact, we can take $M = (Q, \Sigma, \Lambda, q_0, \partial, F)$ where $Q = \{q_0, q_1\}$, $\Lambda = \Sigma \cup V$, $F = \{q_1\}$, and ∂ contains transitions of the forms

1. $((q_0, \varepsilon, \varepsilon), (q_1, S))$; — push the start symbol on the stack
2. $((q_1, \sigma, \sigma), (q_1, \varepsilon))$, for $\sigma \in \Sigma$; and — read a terminal symbol, matching it with the top of the stack
3. $((q_1, \varepsilon, A), (q_1, x))$, for each production $A \longrightarrow x$ in G. — replace a non-terminal on the top of the stack with the right side of the production

1. $((q_0, \varepsilon, \varepsilon), (q_1, S))$; — push the start symbol on the stack
2. $((q_1, \sigma, \sigma), (q_1, \varepsilon))$, for $\sigma \in \Sigma$; and — read a terminal symbol, matching it with the top of the stack
3. $((q_1, \varepsilon, A), (q_1, x))$, for each production $A \longrightarrow x$ in G. — replace a non-terminal on the top of the stack with the right side of the production

$S \longrightarrow AB$
$A \longrightarrow aAb$
$A \longrightarrow \varepsilon$
$B \longrightarrow bB$
$B \longrightarrow b$

$((q_0, \varepsilon, \varepsilon), (q_1, S))$
$((q_1, a, a), (q_1, \varepsilon))$
$((q_1, b, b), (q_1, \varepsilon))$
$((q_1, \varepsilon, S), (q_1, AB))$
$((q_1, \varepsilon, A), (q_1, aAb))$
$((q_1, \varepsilon, A), (q_1, \varepsilon))$
$((q_1, \varepsilon, B), (q_1, bB))$
$((q_1, \varepsilon, B), (q_1, b))$

input: aabbbb

Transition	Input Consumed	Stack (← top)	Derivation
$((q_0, \varepsilon, \varepsilon), (q_1, S))$		S	
$((q_1, \varepsilon, S), (q_1, AB))$		AB	$S \Longrightarrow AB$
$((q_1, \varepsilon, A), (q_1, aAb))$		$aAbB$	$\Longrightarrow aAbB$
$((q_1, a, a), (q_1, \varepsilon))$	a	AbB	
$((q_1, \varepsilon, A), (q_1, aAb))$	a	$aAbbB$	$\Longrightarrow aaAbbB$
$((q_1, a, a), (q_1, \varepsilon))$	aa	$AbbB$	
$((q_1, \varepsilon, A), (q_1, \varepsilon))$	aa	bbB	$\Longrightarrow aabbB$
$((q_1, b, b), (q_1, \varepsilon))$	aab	bB	
$((q_1, b, b), (q_1, \varepsilon))$	$aabb$	B	
$((q_1, \varepsilon, B), (q_1, bB))$	$aabb$	bB	$\Longrightarrow aabbbB$
$((q_1, b), (q_1, b, \varepsilon))$	$aabbb$	B	
$((q_1, \varepsilon, B), (q_1, b))$	$aabbb$	b	$\Longrightarrow aabbbb$
$((q_1, b, b), (q_1, \varepsilon))$	$aabbbb$		

7. Let $M = (Q, \Sigma, \Lambda, q_0, \partial, F)$ be a pushdown automaton. Define $L'(M)$ to be the language $L'(M) = \{w \in \Sigma^* \mid$ it is possible for M to start in state q_0, read all of w, and end in an accepting state$\}$. $L'(M)$ differs from $L(M)$ in that for $w \in L'(M)$, we do not require that the stack be empty at the end of the computation.

a) Show that there is a pushdown automaton M' such that $L(M') = L'(M)$.

b) Show that a language L is context-free if and only if there is a pushdown automaton M such that $L = L'(M)$.

c) Identify the language $L'(M)$ for each of the automata in Exercise 1.

8. Let L be a regular language over an alphabet Σ, and let K be a context-free language over the same alphabet. Let $M = (Q, \Sigma, q_0, \delta, F)$ be a DFA that accepts L, and let $N = (P, \Sigma, \Lambda, p_0, \partial, E))$ be a pushdown automaton that accepts K. Show that the language $L \cap K$ is context-free by constructing a pushdown automaton that accepts $L \cap K$. The pushdown automaton can be constructed as a "cross product" of M and N in which the set of states is $Q \times P$. The construction is analogous to the proof that the intersection of two regular languages is regular, as outlined in Exercise 3.6.7.

Non-Context-Free Languages

- just like there are languages that aren't regular, there are languages that aren't context-free
- there is a pumping lemma for context-free languages similar to the one for regular languages
  - idea for regular languages
    - a DFA/NFA only has a finite number of states, so for any sufficiently long string, a state must be repeated – that provides a loop which can be "pumped" to generate a set of strings, all of which are in the language
  - idea for context-free languages
    - a context-free grammar only has a finite number of non-terminals, so for any sufficiently long string, there's a root-to-leaf path in the parse tree where a non-terminal is repeated – that provides a loop which can be "pumped" to generate a set of strings, all of which are in the language
- the intersection of two context-free languages is not necessarily context-free

General Grammars

Definition 4.6. A ***grammar*** is a 4-tuple (V, Σ, P, S), where:

1. V is a finite set of symbols. The elements of V are the non-terminal symbols of the grammar.
2. Σ is a finite set of symbols such that $V \cap \Sigma = \emptyset$. The elements of Σ are the terminal symbols of the grammar.
3. P is a set of production rules. Each rule is of the form $u \longrightarrow x$ where u and x are strings in $(V \cup \Sigma)^*$ and u contains at least one symbol from V.
4. $S \in V$. S is the start symbol of the grammar.

- a context-free grammar is a grammar where the rules are limited to the form $A \rightarrow x$ (a single non-terminal on the left)

General Grammars

- (general) grammars are more powerful than context-free grammars
- are there languages that can't be produced by grammars?
  - yes – for an alphabet Σ, there uncountably many languages but only countably many that can be generated by grammars

1. Find a derivation for the string *caabcb*, according to the first example grammar in this section. Find a derivation for the string *aabbcc*, according to the second example grammar in this section. Find a derivation for the string *aaaa*, according to the third example grammar in this section.

$$\begin{aligned} S &\longrightarrow SABC \\ S &\longrightarrow \varepsilon \\ AB &\longrightarrow BA \\ BA &\longrightarrow AB \\ AC &\longrightarrow CA \\ CA &\longrightarrow AC \\ BC &\longrightarrow CB \\ CB &\longrightarrow BC \\ A &\longrightarrow a \\ B &\longrightarrow b \\ C &\longrightarrow c \end{aligned}$$

$n_a(w) = n_b(w) = n_c(w)$

$$\begin{aligned} S &\longrightarrow SABC \\ S &\longrightarrow X \\ BA &\longrightarrow AB \\ CA &\longrightarrow AC \\ CB &\longrightarrow BC \\ XA &\longrightarrow aX \\ X &\longrightarrow Y \\ YB &\longrightarrow bY \\ Y &\longrightarrow Z \\ ZC &\longrightarrow cZ \\ Z &\longrightarrow \varepsilon \end{aligned}$$

$a^n b^n c^n$

$$\begin{aligned} S &\longrightarrow DTE \\ T &\longrightarrow BTA \\ T &\longrightarrow \varepsilon \\ BA &\longrightarrow AaB \\ Ba &\longrightarrow aB \\ BE &\longrightarrow E \\ DA &\longrightarrow D \\ Da &\longrightarrow aD \\ DE &\longrightarrow \varepsilon \end{aligned}$$

a^{n^2}

2. Consider the third sample grammar from this section, which generates the language $\{a^{n^2} \mid n \in \mathbb{N}\}$. Is the non-terminal symbol D necessary in this grammar? What if the first rule of the grammar were replaced by $S \longrightarrow TE$ and the last three rules were replaced by $A \longrightarrow \varepsilon$ and $E \longrightarrow \varepsilon$? Would the resulting grammar still generate the same language? Why or why not?

$$\begin{aligned} S &\longrightarrow DTE \\ T &\longrightarrow BTA \\ T &\longrightarrow \varepsilon \\ BA &\longrightarrow AaB \\ Ba &\longrightarrow aB \\ BE &\longrightarrow E \\ DA &\longrightarrow D \\ Da &\longrightarrow aD \\ DE &\longrightarrow \varepsilon \end{aligned}$$

3. Find a grammar that generates the language $L = \{w \in \{a, b, c, d\}^* \mid n_a(w) = n_b(w) = n_c(w) = n_d(w)\}$. Let Σ be any alphabet. Argue that the language $\{w \in \Sigma^* \mid$ all symbols in Σ occur equally often in $w\}$ can be generated by a grammar.

4. For each of the following languages, find a grammar that generates the language. In each case, explain how your grammar works.

a) $\{a^n b^n c^n d^n \mid n \in \mathbb{N}\}$

b) $\{a^n b^m c^{nm} \mid n \in \mathbb{N} \text{ and } m \in \mathbb{N}\}$

c) $\{ww \mid w \in \{a, b\}^*\}$

d) $\{www \mid w \in \{a, b\}^*\}$

e) $\{a^{2^n} \mid n \in \mathbb{N}\}$

f) $\{w \in \{a, b, c\}^* \mid n_a(w) > n_b(w) > n_c(w)\}$