

Pushdown Automata and Context-Free Grammars

1 2 3 4 5 6 7 8 9 10 11 12 13 14



Pushdown Automata and Context-Free Grammars

Equivalence of Pushdown Automata and Context-Free Grammars

Theorem: The class of languages accepted by pushdown automata is exactly the class of context-free languages.

Proving this theorem is not simple. The theorem can be written as two lemmas:

Lemma 1: If a language is a context-free language then it is accepted by some pushdown automaton.

Lemma 2: If a language is accepted by a pushdown automaton then it is a context-free language.

We will examine the first of these, which is the easier part to prove.

1 2 3 4 5 6 7 8 9 10 11 12 13 14

Pushdown Automata and Context-Free Grammars

Chomsky Normal Form

Before we get to the proof of the first lemma, let's sidestep to examine a standardized form for context-free grammars.

A context-free grammar is in **Chomsky Normal Form (CNF)** if

1. the start symbol S does not appear on the right-hand side of any rule,
2. the only possible rule with Λ on the right-hand side is $S \rightarrow \Lambda$ where S is the start symbol,
3. all other rules have the form $A \rightarrow BC$ or $A \rightarrow a$; i.e., each nonterminal yields exactly two nonterminals or a single terminal.

1 2 3 4 5 6 7 8 9 10 11 12 13 14

Pushdown Automata and Context-Free Grammars

Chomsky Normal Form

Theorem: Any context-free language can be generated by a context-free grammar in Chomsky Normal Form.

Proving this is not too difficult, and the proof itself is constructive.

1. Create a rule with a new start symbol that will be used nowhere else.
2. Remove each $A \rightarrow \Lambda$ rule, patching up other rules so that the same language is generated.
3. Remove each $A \rightarrow B$ rule, patching up other rules as necessary.
4. Introduce new nonterminals as necessary to rewrite all rules with exactly one terminal or two nonterminals on the right-hand side.

1 2 3 4 5 6 7 8 9 10 11 12 13 14

Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

Chomsky Normal Form Example

Start with this grammar:	New start symbol	Remove rules with Λ	Remove rules like $A \rightarrow B$	Introduce nonterminals for each terminal
$S \rightarrow ABcC$	$T \rightarrow S$	$T \rightarrow S$	$T \rightarrow ABcC$	$T \rightarrow \bar{a}$
$A \rightarrow AB \mid a$	$S \rightarrow ABcC$	$S \rightarrow ABcC$	$T \rightarrow AcC$	$T \rightarrow \bar{b}$
$B \rightarrow b \mid \Lambda$	$A \rightarrow AB \mid a$	$S \rightarrow AcC$	$T \rightarrow ABc$	$T \rightarrow \bar{c}$
$C \rightarrow D \mid \Lambda$	$B \rightarrow b \mid \Lambda$	$S \rightarrow ABC$	$T \rightarrow Ac$	$T \rightarrow \bar{d}$
$D \rightarrow d$	$C \rightarrow D \mid \Lambda$	$S \rightarrow Ac$	$A \rightarrow AB$	
	$D \rightarrow d$	$A \rightarrow AB$	$A \rightarrow a$	
		$A \rightarrow A$	$B \rightarrow b$	
		$A \rightarrow a$	$C \rightarrow d$	
		$B \rightarrow b$		
		$C \rightarrow D$		
		$D \rightarrow d$		

1 2 3 4 5 6 7 8 9 10 11 12 13 14

Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

Theorem: The class of languages generated by context-free grammars in Chomsky Normal Form is contained in the class of languages accepted by pushdown automata.

Proof: Once again our proof is constructive. We begin by noting the special form of the grammar. During one step of a derivation a nonterminal is replaced either by two nonterminals or by a single terminal.

Further, assuming a leftmost derivation, all terminals will be introduced to the left of all nonterminals.

1 2 3 4 5 6 7 8 9 10 11 12 13 14

Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

Chomsky Normal Form

(Same as on last slide)	Simplify and introduce nonterminals representing pairs of nonterminals	Repeat last step as necessary
$T \rightarrow ABXC$	$T \rightarrow ABY$	$T \rightarrow ZY$
$T \rightarrow AXC$	$T \rightarrow AY$	$T \rightarrow AY$
$T \rightarrow ABX$	$T \rightarrow ABX$	$T \rightarrow ZX$
$T \rightarrow AX$	$T \rightarrow AX$	$T \rightarrow AX$
$A \rightarrow AB$	$A \rightarrow AB$	$A \rightarrow AB$
$A \rightarrow a$	$Y \rightarrow XC$	$Y \rightarrow XC$
$B \rightarrow b$	$A \rightarrow a$	$Z \rightarrow AB$
$C \rightarrow d$	$B \rightarrow b$	$A \rightarrow a$
$X \rightarrow c$	$C \rightarrow d$	$B \rightarrow b$
	$X \rightarrow c$	$C \rightarrow d$
		$X \rightarrow c$

1 2 3 4 5 6 7 8 9 10 11 12 13 14

Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

Assume that G is a context-free grammar in CNF. Let $M = (K, \Sigma, \Gamma, \Delta, s, F)$ be a pushdown automata with $K = \{s, m, f, n_i\}$ where $i = 1, 2, \dots$, (number of rules of the form $A \rightarrow BC$ in G) and $F = \{f\}$. Let Γ be the set of nonterminals in G and let Δ have entries as follows:

- $(s, \Lambda, \Lambda) \rightarrow (m, S)$
- For each rule of the form $A \rightarrow BC$ in G , Δ has the pair of rules
 - $(m, \Lambda, A) \rightarrow (n_i, C)$
 - $(n_i, \Lambda, \Lambda) \rightarrow (m, B)$
- For each rule of the form $A \rightarrow a$ in G , Δ has the rule $(m, a, A) \rightarrow (m, \Lambda)$.
- $(m, \Lambda, \Lambda) \rightarrow (f, \Lambda)$

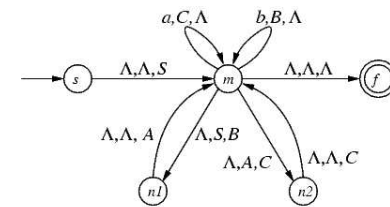
Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

An example here will make the rest of the proof easier to understand.

Consider the context-free grammar $G = (V, \Sigma, R, S)$ where $V = \{S, A, B, C, a, b\}$, $\Sigma = \{a, b\}$ and R is given by the rules on the left below. A transition diagram for the PDA M that would be constructed according the algorithm presented here is on the right below:

$S \rightarrow AB$
 $A \rightarrow CC$
 $B \rightarrow b$
 $C \rightarrow a$



Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

Construct the derivation of aab (the only string in the language of this grammar!) and simultaneously trace it through the PDA.

	$(s, aab, \Lambda) \vdash (m, aab, S)$
$S \Rightarrow AB$	$\vdash (n_1, aab, B)$
	$\vdash (m, aab, AB)$
$\Rightarrow CCB$	$\vdash (n_2, aab, CB)$
	$\vdash (m, aab, CCB)$
$\Rightarrow aCB$	$\vdash (m, ab, CB)$
$\Rightarrow aaB$	$\vdash (m, b, B)$
$\Rightarrow aab$	$\vdash (m, \Lambda, \Lambda)$

Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

To see that M will accept the language generated by G we assume that a string w is in $L(G)$. As we derive w using the rules for G we can trace w through M and see that M will end in an accepting state with an empty stack.

Before doing anything else M pushes S onto the stack.

Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

As the leftmost derivation $S \Rightarrow w$ begins, each time a nonterminal is replaced, the corresponding transition(s) occur in M :

- When a nonterminal is replaced by a terminal in the derivation, M will pop the corresponding nonterminal off the stack and consume the corresponding terminal from the input tape.
- When a nonterminal is replaced by a pair of nonterminals, the old nonterminal is popped off the stack and the two new nonterminals are pushed on the stack in reverse order, so that the first one in the rule is on the top of the stack.

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#)

Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

Because the grammar is in CNF and we are doing a leftmost derivation, the last terminal to be introduced by the grammar is the final symbol in the string being consumed by the PDA.

All of this together shows that our pushdown automaton M accepts exactly the language generated by G . With the completion of this proof we've finished the proof of our first lemma as well; for each context-free language there is a pushdown automaton that accepts it.

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#)

Pushdown Automata and Context-Free Grammars

[prev](#) | [slides](#) | [next](#)

Thus, in the derivation of w from S , every time a terminal is introduced to the string being derived, a terminal is consumed by the PDA and the nonterminal that derives that terminal is removed from the stack.

Nonterminals get onto the stack when a single nonterminal on the stack is replaced with two nonterminals. Once S is pushed on the stack at the beginning of the PDA's operation, the stack remains nonempty until the final nonterminal on the stack is popped and replaced by a terminal in the derivation.

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#)