

# Context-Free Grammars

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15



# Context-Free Grammars

## Language Generators

As we've seen, however, there are languages that are not regular, so we need something other than regular expressions to describe the strings in them.

Consider the following instruction:

$$S \rightarrow aSb$$

which means "when you see  $S$  replace it with  $aSb$ ."

If we start with  $S$  we can get strings like

$aSb$ ,  $aaSbb$ ,  $aaaSbbb$ ,  $aaaaSbbbb$

we could, of course, continue on for ever.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

# Context-Free Grammars

## Language Generators

Thus far the machines we have considered are **language recognizers**; if  $L$  is a regular language then there is a finite automaton that will either accept or reject a string, accepting it if and only if it is in  $L$ .

We want to turn our attention, at least for the time being, to **language generators**.

Actually, viewed in the proper context, **regular expressions are language generators**; they give us instructions that tell us how to construct strings in a language.

For example, consider the regular expression  $a^*bb^*$ . If we think of this as a recipe for creating strings in a language we can interpret it as "write down zero or more  $a$ 's followed by a single  $b$ , followed by zero or more  $b$ 's."

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

# Context-Free Grammars

## Language Generators

Suppose that we had the option of also replacing  $S$  with  $\Lambda$  so that our set of instructions becomes

$$\begin{aligned} S &\rightarrow aSb \\ S &\rightarrow \Lambda \end{aligned}$$

Now we can, at any point we desire, replace  $S$  with  $\Lambda$  yielding a string of just  $a$ 's and  $b$ 's (or the empty string itself). The possible strings look like

$\Lambda$ ,  $ab$ ,  $aabb$ ,  $aaabbb$ ,  $aaaabbbb$

The language we have generated is  $\{a^n b^n \mid n \geq 0\}$ , which we know to be nonregular.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

## Context-Free Grammars

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

### Definition of a Context-Free Grammar

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

- $V$  is an alphabet,
- $\Sigma \subseteq V$  is the set of **terminals**,
- $R \subseteq (V - \Sigma) \times V^*$  is a finite set of **rules**, and
- $S$  is an element of  $V - \Sigma$  and is the **start symbol**.

The set  $V - \Sigma$  is the set of **nonterminals**. Usually we will use uppercase letters for nonterminals and lowercase letters for terminals.

Strings generated by a context-free grammar are all members of  $\Sigma^*$ .

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#)

## Context-Free Grammars

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

### Examples of Context-Free Grammars

We can make the grammar easier to write by allowing multiple rules from a single nonterminal to be written on a single line as:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aA \mid \Lambda \\ B &\rightarrow Bbb \mid \Lambda \end{aligned}$$

**Question:** What strings are in the language generated by this grammar?

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#)

## Context-Free Grammars

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

### Examples of Context-Free Grammars

Consider the following context-free grammar:

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aA \\ A &\rightarrow \Lambda \\ B &\rightarrow Bbb \\ B &\rightarrow \Lambda \end{aligned}$$

Here  $\Sigma = \{a, b\}$ ,  $V = \{S, A, B, a, b\}$ , and  $R$  is specified by the rules above.

Note that each rule has a single nonterminal on the left and a string of any combination of terminals and nonterminals on the right.

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#)

## Context-Free Grammars

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

### Derivations

Given this grammar, we can **derive** the string  $aabbbbbb$  as follows:

$$S \Rightarrow AB \Rightarrow aAB \Rightarrow aaAB \Rightarrow aaB \Rightarrow aaBbb \Rightarrow aaBbbb \Rightarrow aaBbbbbb \Rightarrow aabbbbbb$$

We call this the **derivation** of  $aabbbbbb$  from  $S$ . This derivation has eight steps.

We can denote a multistep derivation with the symbol  $\Rightarrow^*$ ; thus  $S \Rightarrow^* aabbbbbb$ .

The order in which we did these steps was arbitrary. In this example changing the order would not effect the outcome; for other grammars obtaining the correct final string may depend on the exact sequencing of the steps.

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#)

## Context-Free Grammars

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

### Context-Free Languages and Regular Languages

A language is a **context-free language** if there is a context-free grammar that generates it.

We know that there are context-free languages that are not regular. Are there regular languages that are not context-free?

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#)

## Context-Free Grammars

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

### Context-Free Languages and Regular Languages

Clearly the grammar we have constructed is a context-free grammar. We need to be sure that it produces exactly the language  $L$  that the DFA  $M$  accepts.

First we'll show that every string  $w$  accepted by  $M$  can be derived using  $G$ . Then we'll show that every string that can be derived by  $G$  is accepted by  $M$ .

If  $w$  is accepted by  $M$  then there is a path of transitions through  $M$  ending in an accepting state which we'll call  $Q$ . If we start with  $S$  and use the rules from  $R$  corresponding to these transitions we will end with the string  $wQ$ , i.e.,  $S \Rightarrow^* wQ$ .

By construction we know that the rule  $Q \rightarrow \Lambda$  is in  $R$  so we can use that to finish the derivation with the string  $w$ . Thus, if  $w$  is accepted by  $M$ , it can be generated with  $G$ .

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#)

## Context-Free Grammars

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

### Context-Free Languages and Regular Languages

Assume that  $L$  is a regular language. We know that there is some DFA  $M = (K, \Sigma, \delta, s, F)$  that accepts  $L$ .

Create a context-free grammar  $G$  as follows: label the start state  $s$  in  $M$  as  $S$  and label all other states with nonterminals  $A, B, \dots$ . There will be  $|K|$  nonterminals, including  $S$ .

Let  $V = \{Q \mid Q \text{ is a nonterminal label of a state in } M\} \cup \Sigma$ .

Finally create the set of rules  $R$  as follows:

$$\begin{aligned} N \rightarrow aQ & \quad \text{if } \delta(N, a) = Q, \text{ and} \\ N \rightarrow \Lambda & \quad \text{if } N \text{ is an element of } F. \end{aligned}$$

[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#)

## Context-Free Grammars

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

### Context-Free Languages and Regular Languages

Now suppose that there is a derivation using  $G$  such that  $S \Rightarrow^* w$ .

Because of how  $G$  was constructed, we know that every rule in  $R$  that has a nonterminal in the right-hand-side corresponds to a transition in  $M$ .

When we begin processing  $w$  with  $M$  the first transition taken will be exactly the one that corresponds to the first rule used to create  $w$ . The same thing happens for all subsequent rules. Finally, the last nonterminal is replaced with  $\Lambda$ , which only can occur if the state in  $M$  labeled with that nonterminal is accepting.

Thus we know that any string  $w$  generated using  $G$  will be accepted by  $M$ .

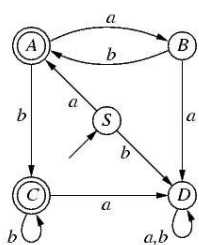
[1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [11](#) [12](#) [13](#) [14](#) [15](#)

## Context-Free Grammars

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

### Context-Free Languages and Regular Languages

For example, consider a DFA that accepts the language defined by the regular expression  $a(ab)^*b^*$ . The states are labeled with nonterminal symbols for the grammar we want to construct.



The rules for our grammar will be

$$\begin{aligned} S &\rightarrow aA \mid bD \\ A &\rightarrow aB \mid bC \mid \Lambda \\ B &\rightarrow aD \mid bA \\ C &\rightarrow aD \mid bC \mid \Lambda \\ D &\rightarrow aD \mid bD \end{aligned}$$

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

## Context-Free Grammars

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

### Context-Free Languages and Regular Languages

Our answer, then, to the question "Are there regular languages that are not context-free?" is "no," because we have shown that given a DFA we can always construct a context-free grammar that generates all strings that the DFA will accept.

Grammars of this sort are called **regular grammars** and always have the form we found here: each rule has  $\Lambda$  or exactly one terminal followed by one nonterminal on the right-hand-side.

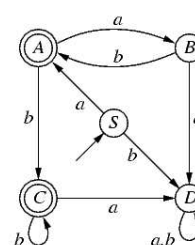
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

## Context-Free Grammars

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

### Context-Free Languages and Regular Languages

If we derive the string  $aababbbb$  using this DFA, and also construct a string using the corresponding rules from our grammar we will end up constructing exactly the string we started with.



$$\begin{aligned} (S, aababbbb) &\vdash (A, ababbbb) & S &\Rightarrow aA \\ &\vdash (B, babbbb) & &\Rightarrow aaB \\ &\vdash (A, abbbb) & &\Rightarrow aabA \\ &\vdash (B, bbbb) & &\Rightarrow aabaB \\ &\vdash (A, bbb) & &\Rightarrow aababA \\ &\vdash (C, bb) & &\Rightarrow aababbC \\ &\vdash (C, b) & &\Rightarrow aababbbC \\ &\vdash (C, \Lambda) & &\Rightarrow aababbbbC \\ & & &\Rightarrow aababbbb \end{aligned}$$

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15