

# Week5 monday

These definitions are on pages 101-102.

| Term  | Typical symbol<br>or Notation       | Meaning  |
|---|-------------------------------------|--|
| <b>Context-free grammar</b> (CFG)                         | $G$                                 | $G = (V, \Sigma, R, S)$  |
| The set of <b>variables</b>                               | $V$                                 | Finite set of symbols that represent phases in production pattern  |
| The set of <b>terminals</b>                               | $\Sigma$                            | Alphabet of symbols of strings generated by CFG<br>$V \cap \Sigma = \emptyset$   |
| The set of <b>rules</b>                                   | $R$                                 | Each rule is $A \rightarrow u$ with $A \in V$ and $u \in (V \cup \Sigma)^*$  |
| The <b>start</b> variable                                 | $S$                                 | Usually on left-hand-side of first/ topmost rule   |
| <b>Derivation</b>   | $S \Rightarrow \dots \Rightarrow w$ | Sequence of substitutions in a CFG (also written $S \Rightarrow^* w$ ). At each step, we can apply one rule to one occurrence of a variable in the current string by substituting that occurrence of the variable with the right-hand-side of the rule. The derivation must end when the current string has only terminals (no variables) because then there are no instances of variables to apply a rule to. |
| Language <b>generated</b> by the context-free grammar $G$ | $L(G)$                              | The set of strings for which there is a derivation in $G$ . Symbolically: $\{w \in \Sigma^* \mid S \Rightarrow^* w\}$ i.e.<br><br>$\{w \in \Sigma^* \mid \text{there is derivation in } G \text{ that ends in } w\}$   |
| <b>Context-free language</b>                              |                                     | A language that is the language generated by some context-free grammar   |

## Examples of context-free grammars, derivations in those grammars, and the languages generated by those grammars

$G_1 = (\{S\}, \{0\}, R, S)$  with rules

$$\begin{aligned} S &\rightarrow 0S \\ S &\rightarrow 0 \end{aligned}$$

In  $L(G_1)$  ...

Not in  $L(G_1)$  ...

$$G_2 = (\{S\}, \{0, 1\}, R, S)$$

$$S \rightarrow 0S \mid 1S \mid \varepsilon$$

In  $L(G_2) \dots$

Not in  $L(G_2) \dots$

$(\{S, T\}, \{0, 1\}, R, S)$  with rules

$$S \rightarrow T1T1T1T$$

$$T \rightarrow 0T \mid 1T \mid \varepsilon$$

In  $L(G_3) \dots$

Not in  $L(G_3) \dots$

$G_4 = (\{A, B\}, \{0, 1\}, R, A)$  with rules

$$A \rightarrow 0A0 \mid 0A1 \mid 1A0 \mid 1A1 \mid 1$$

In  $L(G_4)$  ...

Not in  $L(G_4)$  ...

Design a CFG to generate the language  $\{a^n b^n \mid n \geq 0\}$

*Sample derivation:*

## Week5 wednesday

Warmup: Design a CFG to generate the language  $\{a^i b^j \mid j \geq i \geq 0\}$

*Sample derivation:*

Design a PDA to recognize the language  $\{a^i b^j \mid j \geq i \geq 0\}$

**Theorem 2.20:** A language is generated by some context-free grammar if and only if it is recognized by some push-down automaton.

Definition: a language is called **context-free** if it is the language generated by a context-free grammar. The class of all context-free language over a given alphabet  $\Sigma$  is called **CFL**.

Consequences:

- Quick proof that every regular language is context free
- To prove closure of the class of context-free languages under a given operation, we can choose either of two modes of proof (via CFGs or PDAs) depending on which is easier
- To fully specify a PDA we could give its 6-tuple formal definition or we could give its input alphabet, stack alphabet, and state diagram. An informal description of a PDA is a step-by-step description of how its computations would process input strings; the reader should be able to reconstruct the state diagram or formal definition precisely from such a description. The informal description of a PDA can refer to some common modules or subroutines that are computable by PDAs:
  - PDAs can “test for emptiness of stack” without providing details. *How?* We can always push a special end-of-stack symbol,  $\$,$  at the start, before processing any input, and then use this symbol as a flag.
  - PDAs can “test for end of input” without providing details. *How?* We can transform a PDA to one where accepting states are only those reachable when there are no more input symbols.

Suppose  $L_1$  and  $L_2$  are context-free languages over  $\Sigma$ . **Goal:**  $L_1 \cup L_2$  is also context-free.

*Approach 1: with PDAs*

Let  $M_1 = (Q_1, \Sigma, \Gamma_1, \delta_1, q_1, F_1)$  and  $M_2 = (Q_2, \Sigma, \Gamma_2, \delta_2, q_2, F_2)$  be PDAs with  $L(M_1) = L_1$  and  $L(M_2) = L_2$ .

Define  $M =$

*Approach 2: with CFGs*

Let  $G_1 = (V_1, \Sigma, R_1, S_1)$  and  $G_2 = (V_2, \Sigma, R_2, S_2)$  be CFGs with  $L(G_1) = L_1$  and  $L(G_2) = L_2$ .

Define  $G =$

Suppose  $L_1$  and  $L_2$  are context-free languages over  $\Sigma$ . **Goal:**  $L_1 \circ L_2$  is also context-free.

*Approach 1: with PDAs*

Let  $M_1 = (Q_1, \Sigma, \Gamma_1, \delta_1, q_1, F_1)$  and  $M_2 = (Q_2, \Sigma, \Gamma_2, \delta_2, q_2, F_2)$  be PDAs with  $L(M_1) = L_1$  and  $L(M_2) = L_2$ .

Define  $M =$

*Approach 2: with CFGs*

Let  $G_1 = (V_1, \Sigma, R_1, S_1)$  and  $G_2 = (V_2, \Sigma, R_2, S_2)$  be CFGs with  $L(G_1) = L_1$  and  $L(G_2) = L_2$ .

Define  $G =$



## Summary

Over a fixed alphabet  $\Sigma$ , a language  $L$  is **regular**

- iff it is described by some regular expression
- iff it is recognized by some DFA
- iff it is recognized by some NFA

Over a fixed alphabet  $\Sigma$ , a language  $L$  is **context-free**

- iff it is generated by some CFG
- iff it is recognized by some PDA

**Fact:** Every regular language is a context-free language.

**Fact:** There are context-free languages that are not nonregular.

**Fact:** There are countably many regular languages.

**Fact:** There are countably infinitely many context-free languages.

*Consequence:* Most languages are **not** context-free!

## Examples of non-context-free languages

$$\begin{aligned} &\{a^n b^n c^n \mid 0 \leq n, n \in \mathbb{Z}\} \\ &\{a^i b^j c^k \mid 0 \leq i \leq j \leq k, i \in \mathbb{Z}, j \in \mathbb{Z}, k \in \mathbb{Z}\} \\ &\{ww \mid w \in \{0, 1\}^*\} \end{aligned}$$

(Sipser Ex 2.36, Ex 2.37, 2.38)

There is a Pumping Lemma for CFL that can be used to prove a specific language is non-context-free: If  $A$  is a context-free language, there there is a number  $p$  where, if  $s$  is any string in  $A$  of length at least  $p$ , then  $s$  may be divided into five pieces  $s = uvxyz$  where (1) for each  $i \geq 0$ ,  $uv^i xy^i z \in A$ , (2)  $|uv| > 0$ , (3)  $|vxy| \leq p$ . *We will not go into the details of the proof or application of Pumping Lemma for CFLs this quarter.*