

# CSE 105

# THEORY OF COMPUTATION

---

"Winter" 2018

<http://cseweb.ucsd.edu/classes/wi18/cse105-ab/>

# Today's learning goals

Sipser Section 2.1

- Prove the closure of class of CFLs under certain operations
- Identify when a CFG is ambiguous
- Design a CFG generating a given language

# Context-free grammar

*Sipser Def 2.2, page 102*

$(V, \Sigma, R, S)$

**Variables:** finite set of (usually upper case) variables **V**

**Terminals:** finite set of alphabet symbols  **$\Sigma$**

**Rules/Productions:** finite set of allowed transformations **R**

$$A \rightarrow u \quad A \in V, u \in (V \cup \Sigma)^*$$

**Start variable:** origination of each derivation **S**

The **language generated by a CFG**  $(V, \Sigma, R, S)$  is  $\{ w \text{ in } \Sigma^* \mid \text{starting with the Start variable and applying sequence of rules, can derive } w \text{ on RHS} \}$

# Context-free languages

- $L(00^*)$
- $L((0 \cup 1)^*)$
- $\{abba\}$
- $\{a^n b^n \mid n \geq 0\}$

**Ex:**  $\{0^i 1^j \mid j \geq i \geq 0\}$

recognizable by a PDA

# PDA's and CFGs are equally expressive

**Theorem 2.20:** A language is context-free if and only if some nondeterministic PDA recognizes it.

## *Consequences*

- Quick proof that every regular language is context free
- To prove closure of class of CFLs under a given operation, can choose two modes of proof (via CFGs or PDA's) depending on which is easier

# Closure under union

If  $G_1 = (V_1, \Sigma, R_1, S_1)$  and  $G_2 = (V_2, \Sigma, R_2, S_2)$  are CFGs and  $G_1$  generates  $L_1$ ,  $G_2$  generates  $L_2$ , how can we combine the grammars so we generate  $L_1 \cup L_2$  ?

- A.  $G = (V_1 \cup V_2, \Sigma, R_1 \cup R_2, S_1 \cup S_2)$
- B.  $G = (V_1 \times V_2, \Sigma, R_1 \times R_2, (S_1, S_2) )$
- C.  $G = (V_1 \cup V_2 \cup \{S_0\}, \Sigma, R_1 \cup R_2 \cup \{S_0 \rightarrow S_1, S_0 \rightarrow S_2\}, S_0)$
- D. We might not always be able to: the class of CFG describable languages might not be closed under union.

# Closure under concatenation

If  $G_1 = (V_1, \Sigma, R_1, S_1)$  and  $G_2 = (V_2, \Sigma, R_2, S_2)$  are CFGs and  $G_1$  generates  $L_1$ ,  $G_2$  generates  $L_2$ , how can we combine the grammars so we generate  $L_1 L_2$  ?

# CFGs in the wild

$V = \{E\}$ ,  $\Sigma = \{1, +, x, (, )\}$ ,  $R = \{ E \rightarrow E+E \mid E \times E \mid (E) \mid 1 \}$ ,  $S=E$

Describing well-formed arithmetic expressions

Which of the following strings is generated by this CFG?

- A. E
- B. 11
- C. 1+1x1
- D.  $\epsilon$
- E. I don't know.



# Derivations and parsing

$$E \rightarrow E+E \mid E \times E \mid (E) \mid 1$$

Lots of derivations for  $1+1 \times 1$

$$E \Rightarrow E + E \Rightarrow E + E \times E \Rightarrow 1 + E \times E \Rightarrow 1 + 1 \times E \Rightarrow 1 + 1 \times 1$$

$$E \Rightarrow E \times E \Rightarrow E + E \times E \Rightarrow 1 + E \times E \Rightarrow 1 + 1 \times E \Rightarrow 1 + 1 \times 1$$

$$E \Rightarrow E + E \Rightarrow 1 + E \Rightarrow 1 + E \times E \Rightarrow 1 + 1 \times E \Rightarrow 1 + 1 \times 1$$

# Derivations and parsing

$$E \rightarrow E+E \mid E \times E \mid (E) \mid 1$$

Lots of derivations for  $1+1 \times 1$

$$E \Rightarrow E + E \Rightarrow E + E \times E \Rightarrow 1 + E \times E \Rightarrow 1 + 1 \times E \Rightarrow 1 + 1 \times 1$$

$$E \Rightarrow E \times E \Rightarrow E + E \times E \Rightarrow 1 + E \times E \Rightarrow 1 + 1 \times E \Rightarrow 1 + 1 \times 1$$

$$E \Rightarrow E + E \Rightarrow 1 + E \Rightarrow 1 + E \times E \Rightarrow 1 + 1 \times E \Rightarrow 1 + 1 \times 1$$

# Derivations and parsing

$$E \rightarrow E+E \mid E \times E \mid (E) \mid 1$$

Lots of derivations

$$E \Rightarrow E + E$$

$$E \Rightarrow E \times E$$

$$E \Rightarrow E + E \Rightarrow 1 + E \Rightarrow 1 + E \times E \Rightarrow 1 + 1 \times E \Rightarrow 1 + 1 \times 1$$

A string is **ambiguously derived** in a CFG if it has more than one leftmost derivation i.e. more than one parsing tree

$$1 + 1 \times 1$$

# More complicated language

$$\{a^n b^m \mid n \neq m\}$$

# More complicated language

$$\{a^n b^m \mid n \neq m\}$$

Can rewrite as:

$$\{a^n b^m \mid n > m\} \cup \{a^n b^m \mid n < m\}$$

# More complicated language

$$\{a^n b^m \mid n \neq m\}$$

Can rewrite as:

$$\{a^n b^m \mid n > m\} \cup \{a^n b^m \mid n < m\}$$

$G_1 = ( \{S_1\}, \{a,b\}, R_1, S_1 )$  with rules  $S_1 \rightarrow aS_1 b \mid aS_1 \mid a$

$G_2 = ( \{S_2\}, \{a,b\}, R_2, S_2 )$  with rules  $S_2 \rightarrow aS_2 b \mid aS_2 \mid a$

$G = ( \{S, S_1, S_2\}, \{a,b\}, R, S )$  with rules  $S \rightarrow S_1 \mid S_2, \dots$

# Summary

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
- Context-free** iff it is recognized by some (nondet) PDA
  - iff it is generated by some (ambiguous) CFG

# What's left?

- **Fact:** Every regular language is a context-free language.
- **Fact:** There are context-free languages that are nonregular.
- **Fact:** There are countably infinitely many regular languages.
- **Fact:** There are countably infinitely many context-free languages.

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



# Examples of non-context-free languages

- $\{ a^n b^n c^n \mid 0 \leq n \}$  Sipser Ex 2.36
- $\{ a^i b^j c^k \mid 0 \leq i \leq j \leq k \}$  Sipser Ex 2.37
- $\{ w w \mid w \text{ is in } \{0,1\}^* \}$  Sipser Ex 2.38

*To prove... Pumping lemma for CFLs (won't cover in CSE 105)*

# Closure properties of ...

**The class of regular languages is closed under**

- Union
- Concatenation
- Star
- Complementation
- Intersection
- Difference
- Reversal

**The class of context-free languages is closed under**

- Union
- Concatenation
- Star
- Reversal

**The class of context-free languages is not closed under**

- Intersection
- Complement
- Difference

# For next time

**IndivHW4 due** Tuesday, February 13

For Monday, pre-class reading: pp. 166-167