CSE 105
THEORY OF COMPUTATION

"Winter" 2018

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

Today's learning goals

Sipser Section 2.1

- Design a CFG generating a given language
- Prove the closure of class of CFLs under certain operations
- Identify when a CFG is ambiguous
- Give examples of non-context-free languages

Indiv HW4 due Tuesday

Exam 1 scores released this weekend

Context-free grammar

Sipser Def 2.2, page 102

$(V \cap \Sigma) = \emptyset$

(V, Σ, R, S)

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

Terminals: finite set of alphabet symbols **Σ**

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

$$A \to u \qquad A \in V, u \in (V \cup \Sigma)^*$$

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

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

Context-free languages

- L (00*)
- L((0U1)*)
- { abba }
- { $a^n b^n$ | n ≥ 0 }

Ex: { $0^i 1^j$ | j ≥ i ≥ 0 } recognizable by a PDA

$L(G_1) = \{ w \in \{0,1\}^* \mid$ odd length, 1st, last char is 1 $\}$

$L(G_2) = \{ w \in \{0,1\}^* \mid$ - - - , middle char is 1 $\}$

PDAs 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 PDAs) 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 → __ , A' → __)

A. $G = (V_1 \cup V_2, \Sigma, R_1 \cup R_2, S_1 \cup S_2)$

B. $G = (V_1 \times V_2, \Sigma, R_1 \times R_2, (S_1, S_2))$

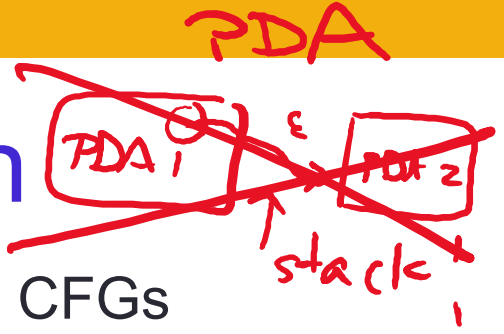
$Q_1 \times Q_2$ DFAs

C.* $G = (V1 \cup V2 \cup \{S0\}, \Sigma, R_1 \cup R_2 \cup \{S0 \rightarrow S1, S0 \rightarrow S2\}, S0)$

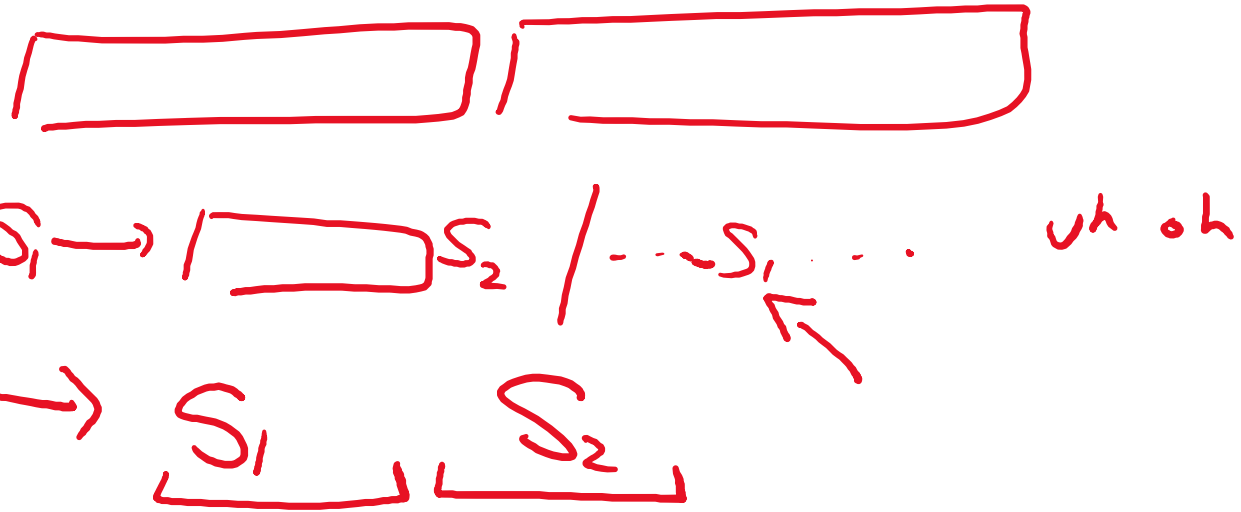
D. We might not always be able to: the class of CFG describable languages might not be closed under union.

* we assume $V_1 \cap V_2 = \emptyset$

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 $L_1 \circ L_2$?



CFGs in the wild

V = {E}, Σ = {1,+,x,(,)}, R = { **E → E+E | ExE | (E) | 1** }, S=E

Describing well-formed arithmetic expressions

Which of the followings strings is generated by this CFG?

A. E not terminal

B. 11 some other symbol

C. 1+1x1

D. ε

E. I don't know.

Derivations and parsing

E → E+E | ExE | (E) | 1

Lots of derivations for 1+1x1

$$E \Longrightarrow E+E \Longrightarrow E+E\times E \Longrightarrow 1+E\times E \Longrightarrow 1+1\times E \Longrightarrow 1+1\times 1$$

$$E \Longrightarrow E\times E \Longrightarrow E+E\times E \Longrightarrow 1+E\times E \Longrightarrow 1+1\times E \Longrightarrow 1+1\times 1$$

$$E \Longrightarrow E+E \Longrightarrow 1+E \Longrightarrow 1+E\times E \Longrightarrow 1+1\times E \Longrightarrow 1+1\times 1$$

Derivations and parsing

2+ 2x2

E → E+E | ExE | (E) | 2 2+2 x2

Lots of derivations for 1+1x1

Not leftmost

Uh oh

$$E \Longrightarrow E+E \Longrightarrow E+E\times E \Longrightarrow 1+E\times E \Longrightarrow 1+1\times E \Longrightarrow 1+1\times 1$$

leftmost

$$E \Longrightarrow E\times E \Longrightarrow E+E\times E \Longrightarrow 1+E\times E \Longrightarrow 1+1\times E \Longrightarrow 1+1\times 1$$

leftmost

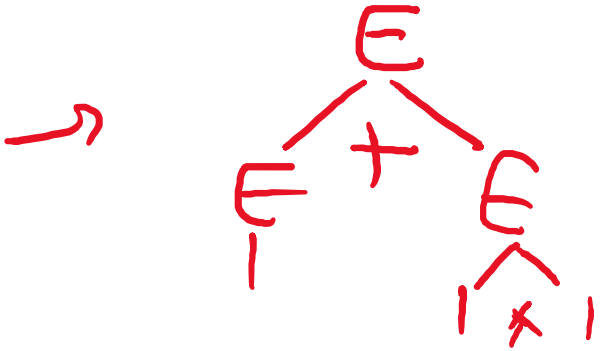
$$E \Longrightarrow E+E \Longrightarrow 1+E \Longrightarrow 1+E\times E \Longrightarrow 1+1\times E \Longrightarrow 1+1\times 1$$

Understanding 1+1x1

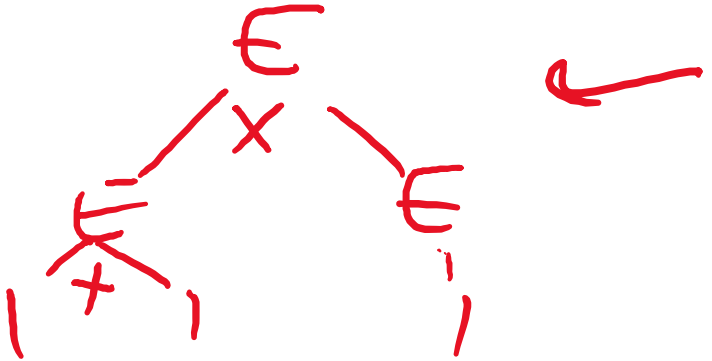
Definition: A derivation is **leftmost** if at every step, the leftmost remaining variable is the one replaced.

Meaningful vs. superficial differences in derivations

Different parsing



Same parsing, different order



Derivations and parsing

E → E+E | ExE | (E) | 1

Lots of derivati

$$E \implies E + E$$

$$E \implies E \times E \qquad + 1 \times 1$$

$$E \implies E + E \implies 1 + E \implies 1 + E \times E \implies 1 + 1 \times E \implies 1 + 1 \times 1$$

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

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:

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

For next time

IndivHW4 due Tuesday, February 13

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