1

Context-Free languages

CSCE 355
2/28/22

Originally studied in the early 1960's to try and understand natural language syntax (Marvin Minski, Noam Chomsky)
Largely unsuccessful (English too complicated)
But useful for describing programming language syntax

Ex: 2 rules (productions)

$$\left.\begin{array}{l} \boxed{S} \to aSb \\ S \to \varepsilon \end{array}\right] \text{Context-free grammar}$$

Start with S

$$\underline{S} \Rightarrow a\underline{S}b \Rightarrow aa\underline{S}bb \Rightarrow aaa\underline{S}bbb \Rightarrow aaabbb$$

derivation (complete derivation)
Can derive all strings of the form $a^n b^n$ $(n \geq 0)$

But $L = \{a^n b^n : n \geq 0\}$ is not regular.

Ex: $S \to (S)S$
$S \to \varepsilon$

$$S \Rightarrow (\underline{S})S \Rightarrow ((S)S)\underline{S} \Rightarrow ((\underline{S})S)(S)S$$
$$\Rightarrow ((\,)\underline{S})(S)S \Rightarrow ((\,))(\underline{S})S \Rightarrow ((\,))(\,)S \Rightarrow ((\,))(\,)$$

Describes language of all well-balanced parentheses.

(2) Derive ()(())

$S \Rightarrow (\underline{S})S \Rightarrow ()\underline{S} \Rightarrow ()(\underline{S})S$

$\Rightarrow ()((S)S)S \Rightarrow \cdots \Rightarrow ()(())$

Leftmost derivation (apply production to the leftmost head in each step.)

If there is any production for a <u>string</u>, there is a leftmost one.

<u>Ex</u>: Grammar for $\{a^n b^m c^n : m,n \geq 0\} =: L$

$S \to aSc$ aabbbcc

$S \to bS$ $S \Rightarrow aSc \Rightarrow aa\underline{S}cc$

$S \to \varepsilon$ $\Rightarrow aabScc \Rightarrow aabbScc$

$\Rightarrow aabbbScc \Rightarrow aabbbcc$

bac: $S \Rightarrow bS \Rightarrow baSc \Rightarrow bac \notin L$

This grammar ~~allow~~ derives strings not in L

Fix so that we get all strings in L <u>and</u> <u>no others</u>.

~~$S \to aST$~~

$S \to aSc$ $T \to bT$

$S \to T$ $T \to \varepsilon$

$S \Rightarrow aSc \Rightarrow aaScc \Rightarrow aaTcc$

$\Rightarrow aabTcc \Rightarrow aabbTcc \Rightarrow aabbbTcc \Rightarrow aabbbcc$

Def: A context-free grammar (CFG) is $\langle V, \Sigma, S, P \rangle$ where

- V is a finite set (elements of V are called nonterminals or variables or syntactic categories)

- Σ is an alphabet such that $\Sigma \cap V = \emptyset$ (elements of Σ are called terminals or tokens)

- $S \in V$ (the start symbol)

- P is a finite set of productions where a production is an expression of the form,

$$A \to \alpha$$

where $A \in V$ (the head of the production) and α is a string over $V \cup \Sigma$ ($\alpha \in (V \cup \Sigma)^*$)

④ Can just list the productions to completely specify a grammar.

$$\boxed{S} \to aSc$$
$$S \to T$$
$$T \to bT$$
$$T \to \varepsilon \quad (T \to \ )$$

- V = set of all the heads ($V = \{S, T\}$)
- conventionally, start symbol is the head of the production listed first.
- Σ = all other symbols in the bodies of the productions (except ε)

Fix a grammar $G = \langle V, \Sigma, S, P \rangle$

Def: Let $\alpha, \beta \in (V \cup \Sigma)^*$. We say α derives β in one step ($\alpha \Rightarrow \beta$) if there exists $A \in V$, $\gamma, \delta \in (V \cup \Sigma)^*$ and a production of the form

$$A \to \zeta \quad (\text{"zeta"})$$

such that

$$\alpha = \gamma \underline{A} \delta$$

and

$$\beta = \gamma \underline{\zeta} \delta$$

(5) A <u>derivation</u> of G is a sequence

$$\alpha_0 \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \cdots \Rightarrow \alpha_n$$

where $n \geq 0$ and $\alpha_i \Rightarrow \alpha_{i+1}$ in one step, for all i such that $0 \leq i < n$.

n is the <u>length</u> of the derivation (# of arrows)

A derivation of G is <u>complete</u> if

$$\alpha_0 \Rightarrow \cdots \Rightarrow \alpha_n$$

$\alpha_0 = S$ and $\alpha_n \in \Sigma^*$ (α_n only terminals)

(a (complete) derivation of α_n).

Say that G <u>derives</u> α if G has a complete derivation of α.

The <u>language of G</u> ($L(G)$) is the language

$$L(G) := \{x : G \text{ derives } x\} \subseteq \Sigma^*$$

<u>Ex</u>: The language of arithmetic expressions using $+, -, *, /$, constants

6 **Def:** A language L is a <u>context-free language</u> (CFL) if L is the language of some CFG.

$E \rightarrow c$

$E \rightarrow E + E$

$E \rightarrow E - E$

$E \rightarrow E * E$

$E \rightarrow E / E$

$E \rightarrow (E)$

(lowercase c ~~reprepresen~~ represents a constant)

derive:

$c * \underline{(c+c)}$

$E \Rightarrow \underline{E} * E \Rightarrow \cancel{c*(E+E)}\ c * \underline{E}$

$\Rightarrow c * (\underline{E}) \Rightarrow c*(E + E)$

$\Rightarrow c*(c + E) \Rightarrow c*(c+c)$ ✓

$c * c + c$

Find 2 <u>different</u> leftmost derivations of $c * c + c$