Theorey of Computation Notes

Yuxuan Sun

Spring 2022

Contents

1 regular languages **2**

2 Non-regular languages **2**

3 Context-Free Grammars **3**

4 Push-Down Autonoma **4**

1 regular languages

Definition 1.1: regular language

language is called a regular language if some finite automaton recognizes it.

2 Non-regular languages

Theorem 2.1: Pumping Lemma

If L is a regular language, then there is a positive integer n (typically, n is the number of states of the DFA accepting L) such that, if $x \in L$ and $|x| \geq n$, then there exist $u, v, w \in \Sigma^*$ such that $x = uvw$ and:

1. $|uv| \leq n$
2. $|u| > 0$
3. for each integer $m \geq 0$, $uv^m w \in L$

Corollary 2.2: infinite

Let the regular language L be accepted by a DFA with n states. Then L is infinite if and only if there is $x \in L$ s.t. $n \leq |x| < 2n$.

Theorem 2.3: Complement

The class of regular languages is closed under complement.

Definition 2.4: Indistinguishable Strings

Let L a language over Σ and let $x, y \in \Sigma^*$. We say that x and y are indistinguishable with respect to L and we write $x \approx_L y$ if, for all $z \in \Sigma^*$, either both xz and $yz \in L$ or neither is. Furthermore, $\approx_L$ can be proved to be an equivalence relation on Σ^*.

Theorem 2.5: Myhill-Nerode

A language L is regular if and only if the number of equivalence classes of $\approx_L$ is finite.

3 Context-Free Grammars

Definition 3.1: Context-Free Grammars

A **context-free** grammar is a 4-tuple $G = (V, \Sigma, S, P)$s.t.:

1. V is a finite set of variables, $S \in V$ is the start variable
2. Σ is a finite set of terminal symbols or teminals s.t. $V \cap \Sigma = \varnothing$
3. P is a finite seet, whose elements are **grammar rules** or **productions** in the form
$$A \to \alpha$$
where $A \in V$ and $\alpha \in (V \cup \Sigma)^*$

Definition 3.2: The Language Generated by a CFG

If $G = (V, \Sigma, S, P)$ is a CFG, the language generated by G is

$$L(G) = \{x \in \Sigma^* \mid S \Longrightarrow_G^* x\}$$

Language L is a context-free language if there is a CFG G s.t. $L = L(G)$

Definition 3.3: derivation tree/parse tree

Let G be a CFG. The **derivation tree** for G is an ordered tree s.t.

1. the root is labeled S
2. every leaf has a label from $\Sigma \cup \{\epsilon\}$
3. every interior vertex has a label from V
4. if a vertex has label A, and its children are labeled (left to right) $a_1, a_2, \ldots, a_n$, where $a_j \in V \cup \Sigma \cup \{\epsilon\}$ for $j = 1, 2, 3, \ldots, n$, then P contains a production of the form $P \to a_1 a_2 \ldots a_n$

Definition 3.4: yield of a tree

the string of terminals obtained by reading the leaves of the tree from left to right, omitting any ϵ.

Theorem 3.5: derivation tree and yield

If G is a CFG, then, for every $x \in L(G)$, there exists a derivation tree of G whose yield is x . Conversely, the yield of any derivation tree is in $L(G)$

Definition 3.6: leftmost derivation

A derivation in a CFG is a leftmost derivation if, at each step, a production is applied to the leftmost variable-occurrence in the current string.

Theorem 3.7: equivalent statement of $x \in L(G)$

1. x has more than one derivation tree
2. x has more than one leftmost derivation
3. x has more than one rightmost derivation

Definition 3.8: ambiguous CFG

A CFG G is ambiguous if, there exists $x \in L(G)$ s.t. x has more than one derivation tree.

4 Push-Down Autonoma

Definition 4.1: PDA

$M = (Q, \Sigma, \Gamma, \delta, q_0, Z, F)$ where

- Q is a finite set of states
- Σ is a finite set which is called the input alphabet
- Γ is a finite set which is called the stack alphabet
- δ is a finite subset of $Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$, the transition relation
- $q_0 \in Q$ is the start state
- $Z \in \Gamma$ is the initial stack symbol
- $F \subseteq Q$ is the set of accepting states